NAVAL POSTGRADUATE SCHOOL
MONTEREY, CALIFORNIA

THESIS

EVALUATING COLOR FUSED IMAGE PERFORMANCE ESTIMATORS

by

James S. Ogawa

September, 1997

Thesis Advisor: William K. Krebs
Second Reader: James N. Eagle

Approved for public release; distribution is unlimited.

DTIC QUALITY INSPECTED 3

19980414 036

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE September 1997	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE
EVALUATING COLOR FUSED IMAGE PERFORMANCE ESTIMATORS

5. FUNDING NUMBERS

6. AUTHOR(S) James S. Ogawa

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Lockheed Martin Corporation, Orlando FL
Office of Naval Research, Arlington, VA

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

This thesis evaluated the effectiveness of sensor fusion—combining infrared and low-light-level imagery—to improve the F/A-18 target standoff range requirement. Several human performance studies have shown inconsistent results regarding the benefits of color-fused imagery. One method to test the validity of sensor fusion is to use mathematical models that simulate and predict the detection abilities of the human visual system. The mathematical models are derived quantitatively from the image statistics, while the behavior data are a qualitative measure of a human observer. This thesis developed a statistical analysis to compare and contrast these techniques to assess sensor fusion. The four models evaluated were: a Global matched filters, a Local matched filter, a Template matching filter, and a contrast-base image quality metric. Of the four models, the Global matched filter produced the highest degree of correlation with the human data. The Global matched filter moderately predicted which of the single-band sensors and which of the fused sensors provided the higher sensitivities despite the characteristically different scenes. Although there are many refinements that need to be explored, the Global matched filter concept may be used to evaluate and compare the many different fusion algorithms being proposed.

14. SUBJECT TERMS
Digital Image Processing, Sensor Fusion, Matched Filters, Image Quality, Image Quality Metrics, Template matching, Target Detection, Receiver-Operator Curves, Contrast Sensitivity Function, Censored Regression.

15. NUMBER OF PAGES
262

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18 298-102

DTIC QUALITY INSPECTED 3

Approved for public release; distribution is unlimited.

EVALUATING COLOR FUSED IMAGE PERFORMANCE ESTIMATORS

James S. Ogawa
Lieutenant, United States Navy
B.S., United States Naval Academy, 1990

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1997

Author: _____
James S. Ogawa

Approved by: _____
William K. Krebs, Thesis Advisor

_____
James N. Eagle, Second Reader

_____
for Richard E. Rosenthal, Chairman
Department of Operations Research

ABSTRACT

This thesis evaluated the effectiveness of sensor fusion—combining infrared and low-light-level imagery—to improve the F/A-18 target standoff range requirement. Several human performance studies have shown inconsistent results regarding the benefits of color-fused imagery. One method to test the validity of sensor fusion is to use mathematical models that simulate and predict the detection abilities of the human visual system. The mathematical models are derived quantitatively from the image statistics, while the behavior data are a qualitative measure of a human observer. This thesis developed a statistical analysis to compare and contrast these techniques to assess sensor fusion. The four models evaluated were: a Global matched filters, a Local matched filter, a Template matching filter, and a contrast-base image quality metric. Of the four models, the Global matched filter produced the highest degree of correlation with the human data. The Global matched filter moderately predicted which of the single-band sensors and which of the fused sensors provided the higher sensitivities despite the characteristically different scenes. Although there are many refinements that need to be explored, the Global matched filter concept may be used to evaluate and compare the many different fusion algorithms being proposed.

TABLE OF CONTENTS

I. INTRODUCTION 1

II. BACKGROUND 5

A. FUSED COLOR ENHANCEMENT 5
B. MATCHED FILTER PERFORMANCE ESTIMATION 8
C. HUMAN FACTORS STUDY TO INVESTIGATE FUSED COLOR IMAGES 12
D. IMAGE QUALITY METRICS (IQM'S) 13
E. SENSITIVITY (d') IN SEARCH DETECTION THEORY 15
F. RESPONSE BIAS (β) IN SEARCH DETECTION THEORY 19
G. HYPOTHESES 20

III. MODEL 23

A. THEORETICAL DEVELOPMENT OF A MATCHED FILTER 23
B. THEORETICAL DEVELOPMENT OF THE CONTRAST-BASED IMAGE QUALITY METRIC 28
C. SETUP OF PLANNED COMPARISONS BASED UPON THE HYPOTHESES 29
D. MODELING A NORMAL DISTRIBUTION FOR CENSORED REGRESSION 30

IV. METHODS 35

A. MATCHED FILTER PROGRAM DEVELOPMENT 35
B. BACKGROUND AND TARGET IMAGERY 36
C. FILTERING AN IMAGE WITH A MATCHED FILTER 38

D. INTERPRETING MATCHED FILTER OUTPUT VALUES 43
E. DEVELOPMENT OF EMPIRICAL RECEIVER-OPERATOR CURVES 46
F. DETERMINATIOIN OF THE HUMAN VISUAL SYSTEM SENSITIVITY AND BIAS 48
G. DETERMINATION OF MATCHED FILTER SENSITIVITY AND BIAS 50
H. OBTAINING THE CSF BASED IQM OUTPUT VALUES 52
I. SETUP OF PLANNED COMPARISONS BASED UPON THE PROPOSED HYPOTHESES 53
J. TREND ANALYSIS OF THE MODELED RESULTS USING CENSORED REGRESSION 54

V. RESULTS 57

A. ANOVA/PLANNED COMPARISON ANALYSIS 66
B. BONFERRONI TESTS FOR COMPARISONS 69
C. TREND ANALYSIS WITH CENSORED DATA 71

VI. CONCLUSIONS 75

APPENDIX A MATCHED FILTER MAIN PROGRAM (MATLAB) 81

APPENDIX B M-FILE TO LOAD IMAGES AND TARGETS 85

APPENDIX C M-FILE TO INPUT FILTERING OPTIONS 93

APPENDIX D M-FILE TO PREPROCESS IMAGES AND/OR TARGETS 99

APPENDIX E M-FILE TO DEVELOP MATCHED FILTER COEFFICIENTS 103

APPENDIX F M-FILE TO DEVELOP A TEMPLATE MATCHING FILTER 111

APPENDIX G M-FILE TO SEQUENTIALLY FILTER THE IMAGE 115

APPENDIX H M-FILE TO FILTER SEVERAL POSITIONS AT A ONCE 121

APPENDIX I M-FILE TO DISPLAY RESULTS AS A ROC PLOT 129

APPENDIX J M-FILE TO DISPLAY DATA 135

APPENDIX K S-PLUS FUNCTION TO PRODUCE SENSITIVITIES 153

APPENDIX L C++ PROGRAM FOR SPECTRAL REWEIGHTING 157

APPENDIX M C++ PROGRAM TO SPLIT IMAGES INTO RGB IMAGES 161

APPENDIX N COLLECTED DATA 163

APPENDIX O EMPIRICALLY-DERIVED ROC PLOTS 199

APPENDIX P RESPONSE BIAS SUMMARY FOR EACH FILTER 205

APPENDIX Q ROC PLOTS OF SENSITIVITY RESULTS 207

APPENDIX R MATLAB FUNCTION FOR CENSORED REGRESSION 225

APPENDIX S OUTPUT OF CENSORED REGRESSION FUNCTION 227

LIST OF REFERENCES 241

INITIAL DISTRIBUTION LIST 245

EXECUTIVE SUMMARY

This thesis evaluates the effectiveness of sensor fusion—combining infrared and low-light-level imagery—to improve the F/A-18 targeting standoff range requirement. The Chief of Naval Operations Air Warfare Division (N-88) endorsed this study to determine if sensor fusion will improve targets' definition and clarity, and, in turn, pilot recognition. Information theory states that displaying two separate sensors as color differences should increase the pilot's ability to discriminate the target from the background. However, several human performance studies have shown inconsistent results. These studies relied on a limited set of stimuli, which may have confounded the results. An alternative method to test the validity of sensor fusion is to use a two-dimensional spatio-spectral matched filter to simulate the behavior of the human operator. The matched filter is derived quantitatively from the image statistics, while the behavioral data are a qualitative measure of an observer. Both techniques have been used to evaluate sensor fusion; however no study has compared the results between the two methods. As a consequence, the results are limited within each discipline and cannot be generalized to the other discipline. This thesis develops a statistical analysis that can be used to compare and contrast the quantitative and qualitative techniques to assess sensor fusion. In addition, this study develops a two-dimensional spatio-spectral matched filter model to predict an observer's ability to detect a target within a natural scene.

Scribner, Satyshur, and Kruer (1993) proposed that combining images from two separate sensors into a single "enhanced" image significantly improve users' target detection and recognition. Objects viewed by low-light and infrared sensors will generally

have the same spatial characteristics, but appear to have dramatically different contrast levels. For example, a pseudo-color target has better scene contrast (e.g., the horizon or a tree line is much easier to segment), which should attract the pilot's attention, thereby reducing the pilot's workload. This is analogous to a ground soldier laser-designating a target so that the pilot can easily identify its location. Accordingly, color fusion, as compared to monochrome devices, will allow users to quickly orient themselves with respect to a scene and detect targets with greater accuracy.

The intent of the matched filter technique was to model CAPT Matthew Sampson's USMC thesis data (1996). Sampson found that aviators' accuracy for detecting a pseudo-color target was equal to that of either the infrared or image intensified target. However, there were significantly different levels of performance across the four sensors for each of the scenes. Aviators could quickly detect a pseudo-color target within one scene, but failed to detect the pseudo-color target in a different scene. The three scenes used for the human factors testing each exhibited characteristically different proportions of emmissivity and reflectivity, which would greatly influence the aviators' detection of a pseudo-color target. This limited set of scenes could account for the wide range of detection ability and the inconclusive results.

A two-dimensional matched filter array can be used to determine the presence or absence of an object within an image. Scribner, Satyshur, and Kruer (1993) used the matched filter to predict an observer's ability to detect a target within a natural scene. Their results showed that a pseudo-color target was superior to single-band infrared targets. The advantage of the matched filter technique is the ability to predict an

observer's detection rate across multiple scenes. The validity of this technique depends on whether the matched filter accurately predicts the detection rate of a human observer.

This thesis tested whether a two-dimensional matched filter or contrast based image quality metric could predict aviators' target detection ability for the following sensors: Infrared, low-light image intensified, combination of low-light and infrared with a monochrome output, and, combination of low-light and infrared with a pseudo-color output. Three still images were used from an early prototype fusion sensor system developed by Texas Instruments and the Night Vision Electronic Sensor Directorate (Palmer, Ryan, Tinkler, and Creswick, 1993). These images were collected from a low-light visible image intensifier (.6-.9μm) and a first-generation forward-looking infrared (8-12μm) mounted on a UH-1N helicopter. Images were collected during starlight (10^{-3} lux) to full moon (10^{-1} lux) conditions.

The matched filter used for this experiment is a spatial filter optimized for a target's signal to noise-power, which is shifted to several locations within an image. At each location, the filter's coefficients are multiplied by the pixels that are overlaid and summed, providing a measure of the correlation between the overlaid scene and the target. By evaluating several background areas of an image with, and then without, a target present, the filter's signal-plus-noise and noise distributions may be estimated. These empirically-derived distributions can be used to estimate the matched filter's ability to "discriminate" the target from the background noise. This ability to "discriminate" is expressed in the d-prime (d') sensitivity convention of search detection theory, facilitating a direct comparison to the sensitivities of the human subjects.

Results of the ANOVA analysis indicated a significant difference between the performance evaluations of the matched filters and the human subjects. However, graphical analysis of the resulting sensitivities showed that the same trends in the human subjects' ability to discriminate the target from the background are also reflected in the sensitivities produced by the matched filter. Although regression analysis showed highly significant regression coefficients, the highest coefficient of determination was for the Global matched filter at 31 percent. However, in almost all cases the matched filters were able to correctly predict which of the sensors provided the worst detection rates.

Since these models are based on the human visual system and should be comparable to the detection rates of an aviator, they can be used to evaluate the various sensor fusion processes. The matched filter has two advantages. First, it quantifies the degree of "enhancement" achieved by a fusion process, thus allowing for direct comparisons of the various sensor fusion techniques. The second advantage is the ability to predict human visual performance under a variety of environmental and terrain conditions.

ACKNOWLEDGMENT

The author would like to acknowledge the financial support of Lockheed Martin Electronics and Missiles, and the Office of Naval Research. This work was performed under Contract N0001497WR30091.

The author wants to thank Prof. William Krebs and Prof. James Eagle for their guidance and patience toward the completion of this thesis. And many thanks to my wife Michelle, whose understanding and patience throughout the course of these studies contributed greatly to its completion.

I. INTRODUCTION

Within the last decade, the conceptual template of the U.S. military's fighting force has dramatically changed. In the past, the United States Armed Forces were built around a core set of capabilities relying on massed forces and sequential operations to handle domestic, regional, or international demands. In the future, the armed forces will need to adapt to a reduced budget while maintaining the capability to adapt to an increasingly dynamic environment. Thus, the military will benefit from a capabilities-based automated information system designed to leverage technological opportunities in order to achieve dominance across the range of military operations. This "information superiority" will allow the Joint U.S. forces to achieve massed effects by concentrating combat power at the decisive time and place with unparalleled precision. This vision is, "built on the premise that modern and emerging technologies— particularly information-specific advances— will have a tremendous impact on the use of military forces" (General John M. Shalikashvili, Chairman, Joint Chiefs of Staff).

A key component of this vision is the concept of fused "all-source" intelligence. Real-time information gathered from all levels will be assembled into a fuller understanding of the situations at hand and disseminated to requisite forces. Putting a fused "full picture" into front line units calls for enhanced sensors and better weapons to extend the range of offensive actions. Under contract by the U.S. Technology Reinvestment Program, FLIR Systems, Inc. is part of a multi-company effort to develop a real-time "fused" image from a normal nighttime scene and infrared imagery *(Aviation Week and Space Technology*, April, 1997). The goal of this program is to develop and

demonstrate a system that would allow a pilot to recognize and maintain visual contact with the runway at night, even in heavy precipitation and fog. Improved warfighting technologies such as this could be extended to allow U.S. forces to engage in warfare operations seamlessly in day and night, good weather or bad, thus extending our range of offensive capabilities. The goal of enhanced vision, particularly at night, is to allow U.S. forces to fly, drive, steam, and fight anywhere, at anytime. (*Aviation Week and Space Technology*, April, 1997)

The challenge of fusing intensified light (IL) and infrared (IR) nighttime imagery into an enhanced image is not unique to FLIR Systems. Several efforts are underway proposing various approaches to sensor fusion. Massachusetts Institute of Technology Lincoln Laboratory (MITLL), Texas Instruments Corporation (TI), and the Naval Research Laboratory (NRL) in conjunction with the Naval Postgraduate School (NPS) have their own methods of fusing these nighttime images. These techniques may differ on the algorithm approach, but they all have the same objective: improving the image quality for the observer. Several behavioral studies (Essock, McCarley, Sinai, and Krebs, 1996; Sampson, Krebs, Scribner, and Essock, 1996; Steele and Perconti, 1997) and image quality studies (Scribner, Satyshur, Kruer, 1993; Waxman, Savoye, Fay, Aguilar, Gove, Carrick, and Racamato, 1996) have tried to quantify the benefits of sensor fusion, but the results were inconsistent. The purpose of this thesis is to model human performance using two mathematical models, a matched filter and a contrast based image quality metric, both of which are based on the human visual system. The advantages of the model approach are twofold. First, it quantifies the degree of "enhancement" achieved by a fusion process,

thus allowing for direct comparisons of the various sensor fusion techniques. The second advantage is the ability to predict human visual performance under a variety of environmental and terrain conditions.

II. BACKGROUND

A. FUSED COLOR ENHANCEMENT

Intensified-light and long-wave infrared sensors are the primary sensors for military night operations. These sensors provide a unique view of the scene being observed due to the differing frequency bands of operation. Scribner, Satyshur, and Kruer (1993) proposed that combining images from two separate sensors into a single "enhanced" image significantly improves user's target detection and situational awareness. It is hypothesized that the probability of missing a target is much greater for a single band sensor than for the composite fused band. In addition, combining multiple bands into a single fused color scene should enhance the contrast between the foreground and background, e.g., separation of vegetation (Scribner, Satyshur, Shuler, and Kruer, 1996).

The hypothesized advantages of pseudo-color enhanced images are based on the human vision system's (HVS) ability to discriminate various frequency bands within visible light. Color perception in the HVS evolved primarily to increase survival by better facilitating detection and discrimination. This is achieved by the photosensitive pigments in the cone visual receptors within the human eye. Normal human vision is trichromatic, which allows the HVS to make use of three "color bands" when observing a daylight scene (Sekuler and Blake, 1990). Certain colors, such as red, have instinctive psychological impact and tends to draw attention, enhancing detection and discrimination of possible dangers. Other animals, such as snakes of the subfamily Crotalinae (e.g., pit-vipers, rattlesnakes, copperheads, and water moccasins) and the Boidae family (e.g., pythons), also make use of more than one "color" band to enhance their survival (Hartline

and Newman, 1982). These snakes have evolved infrared-sensitive pit organs which can sense with deadly accuracy a small , warm object, such as a mouse, at a range of half a meter without visual information. Thus, their ability to combine the infrared signals from the pit organs with visual signals from the eyes further enhances detection and discrimination.

If the HVS were monochromatic, then scenes would look much like an image displayed on a monochrome CRT. Digitized monochrome images physically represent each pixel as one of 256 different levels of brightness (grey-levels). However, the HVS can perceptually distinguish only about 30 grey levels on a CRT with images collected under ideal lighting conditions. For images acquired under very dim light, or some other mode such as image-intensified devices or long-wave infrared sensors, the grey-scale depth of the image is even more limited. Therefore, a pseudo-color enhanced digital image would allow for better classification of various regions or points of specific information within the image. It would also aid the HVS in distinguishing subtle variations which are imperceptible in the single-band monochrome displays (Russ, 1995).

It is important to understand the distinction between a color scene observed through the human eye and a processed pseudo-color scene. As previously mentioned, in a naturally lit environment, various colors have certain psychological impact that has evolved over time. In order to ensure that these color queues appeared the same for a wide range of environs throughout the day, the HVS evolved the ability to distinguish these colors under varying natural lighting conditions (Sekuler and Blake, 1990). For example, a color such as red tends to appear the same shade of red whether indoors or

out, while in bright afternoon sunlight or in a late afternoon sunset. The HVS's ability to adapt to small variations in the spectral distribution of natural light and provide consistent physiological queues is termed color constancy (Sekuler and Blake, 1990). Algorithms exist which allow a television camera to exhibit something similar to color constancy. However, because the intensified light images are taken at such low levels of illumination, all colors appear gray. In the case of the infrared images, the infrared band is decidedly outside the range of natural light and cannot provide color queues which correlate with the visible bands. As a result, the intuitively "correct" color information of the objects within any of these scenes is lost. The objective of pseudo-color processing emphasizes color contrasting rather than reproducing the intuitive colors of the scene, as demonstrated by the pseudo-color process developed by NRL. However, with a priori information as to what these intuitive colors should be, it is possible to provide a more intuitive pseudo-colored scene, as demonstrated by MIT's pseudo-color process.

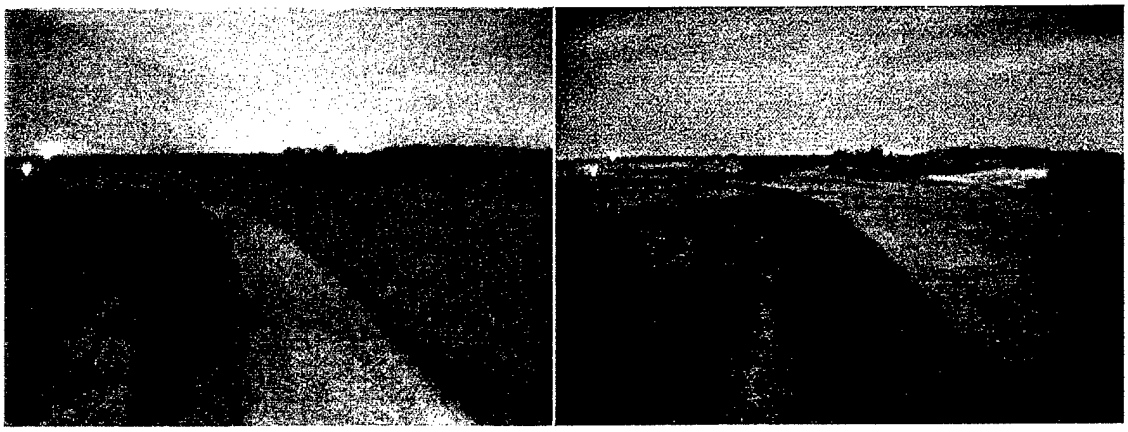


Figure 1. Examples of Color Fusion processing. On the left is NRL's fusion process; on the right is MIT's fusion process using a piori knowledge of scene content. (Courtesy of the U.S. Army Night Vision and Electronic Sensors Directorate).

B. MATCHED FILTER PERFORMANCE ESTIMATION

Statistical signal processing has been widely used in communications and other areas requiring discrimination of a signal from noise. Image detection is the natural extension from one-dimensional vector sampling to a two-dimensional signal, such as a digital image. One image detection technique employs the use of a spatial "matched" filter to determine the presence or absence of an object within an image. A matched filter is a two-dimensional (2-D) array which has been optimized to maximize the signal-to-noise ratio and provide a measure of the spatial correlation between the input image and the reference image (Pratt, 1991). The resulting filters are "tuned" to negate the effects of the background (noise) of the image in which the filter is to be used and enhance the detection of the target. Scribner et al. (1993) used a matched filter on long-wave infrared (9.0 to 11.6 μm), medium-wave infrared (4.5 to 5.5 μm), and short-wave infrared (2.0 to 2.6 μm) sensors, as well as on a fused single image of these bands. In this approach, spatial-only and spatial-spectral matched filters were derived for the three infrared images and the fused composite image respectively. The intent of these matched filters was to simulate the detection ability and sensitivity of the HVS. The single-band filters were derived using the smoothed power spectrum of the background and the proposed target template. The 3-D spatio-spectral (color) filter was derived by considering the target and the background as a 3-D space, with the third dimension being the spectral values at each pixel. The respective filter is then shifted to several locations within the image; the values are multiplied by the pixels that are overlaid; and the summed value is stored for each position, showing where regions identical or similar to the target are located. These values

are then compared to a second set of calculated totals produced by the same procedure, but with the target inserted at corresponding locations. By comparing the signal-plus-noise values to the noise values for a given threshold value, false alarm and target detection probabilities can be calculated and displayed in the form of an empirical receiver-operator curve (ROC) (see Figure 2).

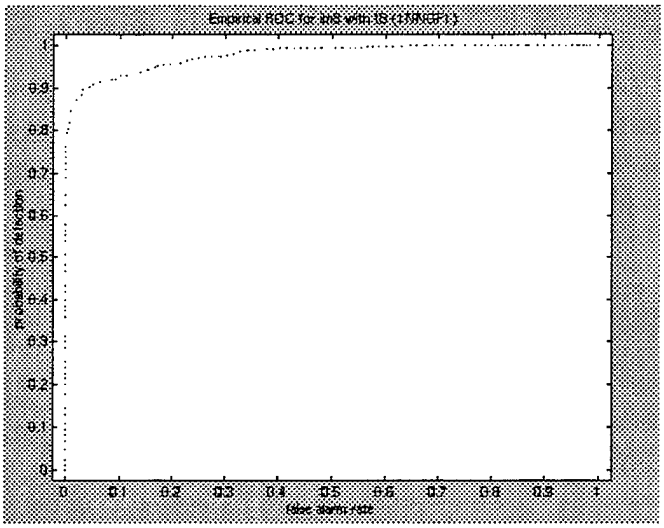


Figure 2. Example of an empirically derived receiver-operator curve (ROC) for the monochrome fused "rectangle" scene using a 2-D global matched filter.

The background of an image can be viewed two ways. One way is to use the entire image (less the target) to match the target filter, resulting in a globally matched filter. Scribner et al. (1993) used this approach on overhead aerial images. Another approach is to use only those areas within the image in which the target sensibly will be located. For example, an image taken from an aircraft at low altitude often contains a horizon separating the sky from land. If the target to be detected is a truck then the sky portion of the image may be removed from consideration. Thus, the background of the land area is used to match the target filter, resulting in a locally matched filter. Since all of the images used for the human factors experiment (Sampson, 1996) were taken from an

aircraft at low altitude, the local matched filter should perform better than a global matched filter because the targets are all objects found on the ground. Thus, a large difference should exist between the performance predictions of the local and global matched filters.

When applied to aerial images of natural scenery, this matched filter approach resulted in as much as a full order of magnitude increase in target detection rate for the "composite" (color fused) images (Scribner et al., 1993). However, the validity of these results hinges on the accuracy with which the matched filter models the sensitivity of the HVS. The primary limitation of the matched filter approach is that the output is the degree of correlation between the "energy" of the input and reference images rather than the correlation between the spatial structure. Thus, any area having the same relative energy would be seen as the target, regardless of the spatial shape. This type of matched filter generally provides poor discrimination between objects of different shapes. However, it will locate regions of similar size and energy (Pratt, 1991). Also, Scribner et al. (1993) used the preprocessing technique of normalization of the background image to standardize the images and remove the variations in lighting conditions since the various images were taken under different conditions. This processing is similar to the removal of the DC component within the image. Removing this DC component prior to filtering with a matched filter should extend the dynamic range of values achieved by the matched filter (Russ, 1995).

A filtering process similar to the matched filter process is referred to as template matching. In this process, the target itself is used as the filter or template. The application

of this filter is carried out the same way as for the matched filter. Like the matched filter, the target template is then shifted to several locations within the image, the values multiplied by the pixels that are overlaid, and the summed value stored for each position, showing where regions identical or similar to the target are located. As the target template passes over a target, it will output a maximum value, whereas other areas without the target will result in relatively low output values. This target template is extremely specific, producing a maximum value only when an exact match in shape and coloring has been found. Therefore, template matching represents one extreme of machine vision since the slightest deviation from the template will be reflected by an inordinately lower value relative to the maximum. However, this process could provide a basis for comparing the performance evaluations produced by the matched filter process since template matching reflects the most specific target search of the image being filtered. As a means of modeling the HVS, this template matching process has numerous short-comings. The theory of a matched filter is theoretically different in that it is optimized to enhance "target-like" detections while reducing the effects of noise by discriminating on the basis of relative "energy". However, the results may still closely follow the trends of an ordinary template matching process. Thus, when comparing the results of the matched filter with the results of template matching, we would expect to see significant differences in performance predictions across the combinations of scenes and sensors.

C. HUMAN FACTORS STUDY TO INVESTIGATE FUSED COLOR IMAGES

Based on the superiority of fusion and color hypothesized by the matched filter approach (Scribner et al., 1993), a human factors experiment (Sampson,1996) at the Naval Postgraduate School investigated a priori belief that the results from a visual search paradigm would favor a color-fused image over an infrared or intensified light image alone. A color-fused, monochrome-fused, LWIR, or intensified light image was displayed (with or without a target) for 600 milliseconds and then removed. Subjects then rapidly responded, indicating whether there was a target in the scene or not, with reaction time and accuracy as the dependent measures. The experiment failed to provide enough evidence to show lower mean reaction times for color-fused or monochrome-fused images when compared to the those for the individual sensor bands (Sampson, 1996).

The non-significant results could be explained by the matched filter technique which is highly dependent on both the specified target colors and the background statistics, which vary immensely from scene to scene. This is especially true for images acquired in low light conditions where statistical fluctuations become important (Russ, 1995). Thus, the application of the same matched filter process to the fused images and individual bands used in the human factors testing are bound to vary greatly. In light of the specific nature of the matched filter approach, the use of image quality metrics based upon models of the HVS may be a more appropriate means of predicting color-fused image performance.

D. IMAGE QUALITY METRICS (IQM'S)

Image quality refers to the detail provided by a visual image, which directly correlates to the depth of information that can be extracted from that image. A low-quality image may provide basic situational awareness as to the type of scene being observed. Examples of low-quality images are those whose depth of detail allow an observer to detect such objects as railroad yards or a ship in a harbor. Higher-quality images would provide increased depth of information, such as being able to identify vehicle types parked at the railroad yard or classification of the ship type and ongoing activities in the harbor (Nill and Bouzas, 1991). Thus, IQM's are much harder to define and measure since "quality" compensates for human perception. One such IQM is the Contrast Sensitivity Function (CSF). The CSF represents what an individual can see across different spatial frequencies. For example, the furthest point (high spatial frequency) represents visual acuity. A CSF is obtained by the visual perceptions of several subjects observing a test "grating" on a specifically designed and calibrated television screen while the contrast was varied by experimenters. Contrast was increased until the test gratings were just visible to each subject and the contrast threshold recorded. This procedure was repeated for test "gratings" of various spatial frequencies, and the results averaged (Sekuler and Blake, 1990). Figure 3 shows the resulting contrast threshold curve separating the region corresponding to contrast levels below the threshold curve (a decrease in contrast), and the pattern not being visible (region above the threshold curve), and the region corresponding to contrast levels above the threshold curve (an increase in contrast), and the pattern becoming visible (region below the threshold curve).

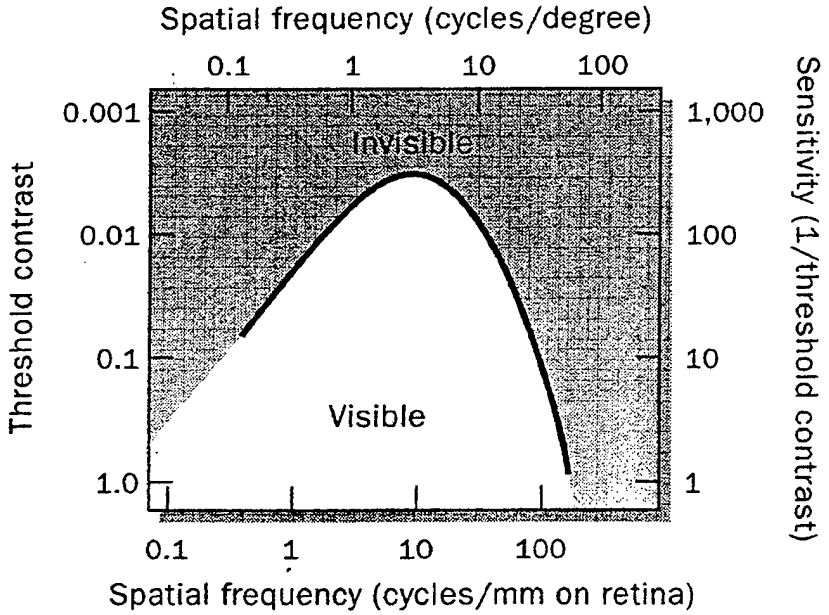


Figure 3. A contrast sensitivity function for an adult human. The upper horizontal axis is scaled in units specifying the number of pairs of light and dark bars of the grating falling within one degree of visual angle on the retina. (Sekuler and Blake, 1990)

A characteristic difference between the CSF and a physical transfer function of a lens is that the CSF drops at lower frequencies, implying that the HVS is more sensitive to intermediate frequencies. Therefore, the contrasting cues available to the HVS provided by a digitized image may be quantitatively measured and compared directly to the performance predicted by the matched filter process and to the results of the human factors testing.

Due to the limitations of the sensing and display media, any image displayed on a CRT is subject to a spatial degradation of the observer's CSF. Knowledge of the CSF's shape may also allow compensation for the display degradation by weighting the intermediate frequencies of a digitized image. A similar approach is done to correct midrange sensitivity degradation in the HVS. Low-vision patients utilize visual aids that

boost midrange spatial frequencies. Of several previously developed models and approximations of the CSF, the generalized model of human contrast sensitivity developed by Pelli, Rubin, and Legge (1986, 1990) implements a parabolic fit of the CSF. This non-linear fit models the input-output representation of the HVS at moderate- to low-level illumination. Thus, weighting the digitized single band images and the color fused images and comparing the resulting CSF IQM's enables us to develop techniques to quantitatively evaluate the performance of the various color fusion methods, as well as to quantify and scale the advantages gained by multi-spectral color fusion.

E. SENSITIVITY (d') IN SEARCH DETECTION THEORY

Whether a target detection is accomplished through the HVS or through electronic means, such as the matched filter, the theory of signal detection requires recognizing a signal from a background of noise. Thus, accomplishing signal detection requires determining the rate at which the signal plus random noise is correctly detected, as well as determining the rate at which the random noise is incorrectly determined to be a signal plus noise (Macmillan and Creelman, 1991). These rates are the probability of detection or "hit" rate (Ph), and the probability of a false alarm (Pfa) respectively. These probability measures allow the calculation of the standard sensitivity of Search Detection Theory, d-prime (d') for an observer viewing the corresponding image (Macmillan and Creelman, 1991). This sensitivity is a measure of the proficiency in which the signal plus noise is discriminated from the noise and is the difference, in units of standard deviations, between the means of the respective distributions (Macmillan and Creelman, 1991). Thus, when a

strong signal is applied to a Gaussian noise background, the resulting difference between the means of the signal plus noise distribution and the noise distribution becomes large. For weaker target signals, the difference between the two distributions decreases until the signal is so weak that the distributions are virtually the same, resulting in significant overlap and great difficulty in signal discrimination (Gescheider, 1985).

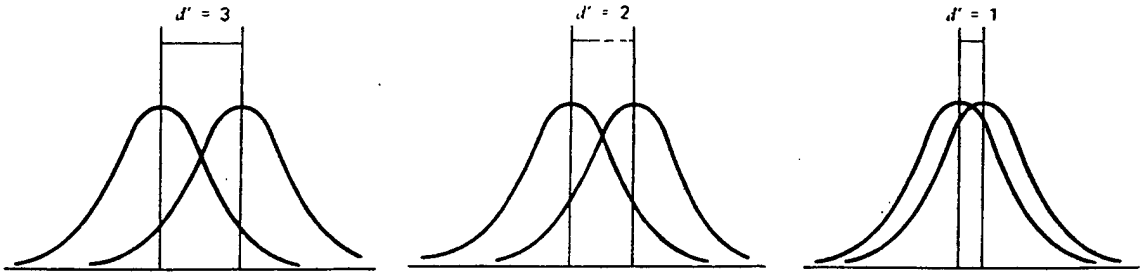


Figure 4. Three representations of paired "Noise" and "Signal+Noise" distributions for d'=3, 2, and 1. This illustrates the measurement of d' as well as the trend from a well discriminated "strong" signal (d'=3) to a relatively "weak" signal (d'=1) (Schiffman, 1990).

This sensitivity is defined in terms of the inverse of the normal distribution, $Z(\cdot)$ (Macmillan and Creelman, 1991):

$$d' = Z(Ph) - Z(Pfa)$$

In order to avoid infinite results, such as in the case of a perfect score of Ph=1.0 and Pfa=0.0, experimenters have accepted an upper probability limit of 0.99 and a lower probability limit of 0.01. Thus, a perfect score would result in an upper bound on d' of 4.65, while a moderate performance implies a d' of 1.0 and a "random" performance having a d' equal to zero (Macmillan and Creelman, 1991).

Calculation of sensitivity from the results of the human factors experiment (Sampson, 1996) is a straightforward proportion of "hits" and "false alarms" to the number of trials presented for each image. Figure 5 shows four scenes from one of the

three images used containing a tower on a grass field background with the target, a satellite dish, in the three positions presented. The fourth picture shows the same scene less the target, which was presented along with the other three randomly as a distracter.

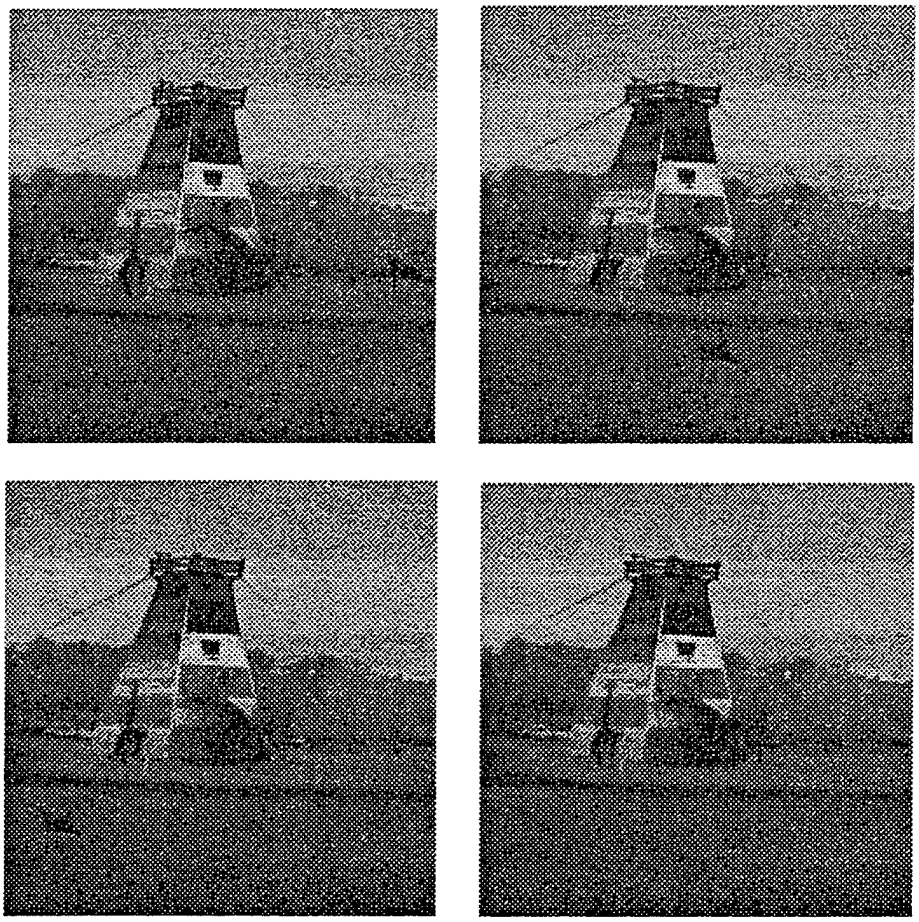


Figure 5. Examples of the four "target positions" presented to subjects of the human factors experiment (Sampson, 1996). This is an example of the intensified light sensor (Courtesy of the U.S. Army Night Vision and Electronic Sensors Directorate).

A correct response is scored as a hit if the observer responds that a target was present, and one of the first three scenes containing the target has just been presented. An incorrect response is scored if the observer responds that a target is present when, in fact, scene four has just been shown. This is counted as a false alarm.

In the case of the matched filter, the signal to noise and noise values for several

points throughout the same images used in the human factors experiment will be computed. The hit and false alarm rates will then be calculated according to local and global thresholds developed from each of the three target positions from each image. At each target position, the signal plus noise (SNt) and noise (Nt) values are computed. These SNt and Nt values are then used as global thresholds to determine the hit and false alarm rates. Any SN value from the several previously calculated and stored values which is greater than Nt will be viewed as a "hit", while any N value from the several previously calculated and stored values which is greater than SNt will be viewed as a "false alarm". For those SN and N pairs of the previously stored values which fail to exceed the global "hit" threshold, Nt, an additional local threshold based on the relative magnitude between the SNt and the Nt, is applied. According to search theory, these values will differ in relative magnitude such that the SNt value at the target position will be greater than the Nt value. This relative magnitude is related to how well the matched filter can distinguish the target signal above the local noise present. Thus, from the values calculated and stored previously, any SN and N pairs with a relative magnitude equal to or greater than the relative magnitude between SNt and Nt is viewed as distinguishing the target from the background and is counted as a hit. Thus, calculating Ph and Pfa becomes a straightforward proportion of "hits" and "false alarms" to the number of points evaluated within each image. Since sensitivity can be determined for the HVS as well as for the matched filter, it will allow a direct comparison of the HVS to the matched filter.

F. RESPONSE BIAS (β) IN SEARCH DETECTION THEORY

In search detection theory, a constant d' value can be obtained by many combinations of Ph and Pfa. A plot of these Ph and Pfa values (for a constant d') results in a ROC plot. Where observers actually fall on this ROC plot is a function of their response bias. This is the tendency for an observer to favor one response over the other (Macmillan and Creelman, 1991) can be varied by the addition of rewards or penalties for correct and incorrect responses. One measure of response bias, β, is a likelihood criterion based on the probability density function (pdf) of the normal distribution (Gescheider, 1985).

$$\beta = \frac{\phi(Ph)}{\phi(Pfa)}, \text{ where } \phi(\cdot) \text{ is the pdf of the Normal distribution.}$$

The value of β corresponds to the point on the ROC plot tangent to a line of slope β. Thus, β equal to one corresponds to no bias and represents the point on the ROC plot that intersects the minor axis of the plot (see Figure 6). The minor axis travels in a straight line from the upper right corner diagonally down to the lower left corner. A value of β greater than one represents a response bias on the lower half of the ROC plot, while a β less than one would be found on the upper half of the ROC plot.

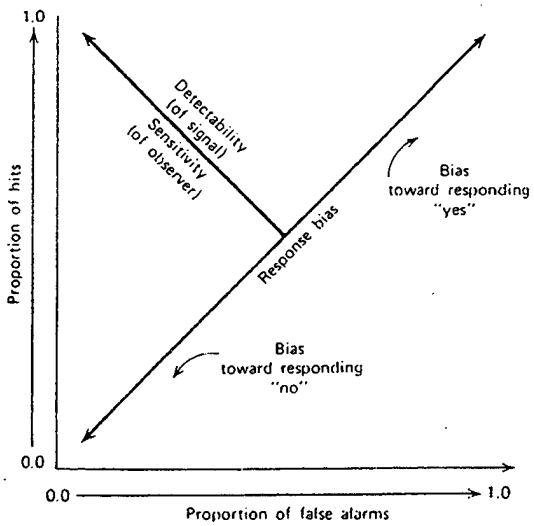


Figure 6. This illustrates the features depicted on a ROC plot. On the major axis is the observer's sensitivity (d'). On the minor axis is the observer's response bias (β) (Schiffman, 1990).

Since β is dependent upon the previously calculated values of Ph and Pfa, it will also be possible to compare the response bias of the HVS to that of the matched filter.

G. HYPOTHESES

The purpose of this thesis is to investigate the validity of using matched filters or psychophysically based image quality metric in evaluating various multi-spectral color fusing algorithms. If the matched filters or psychophysically based image quality metric do not model the HVS then we would expect to see large differences between the sensitivities produced by the models and the sensitivities from the human subjects. Additionally, if the matched filter concept is a viable method of modeling the HVS, then there should be a large difference between the Globally matched filter and the Locally matched filter sensitivities.

The Template matching filter was included in this evaluation since it is a standard image-processing search paradigm which makes no attempt to model the HVS. If the matched filters outputs correlate highly with the results of the Template matching filter rather than the results of the human data then modeling the HVS with the matched filters would be questionable. Thus the hypotheses developed to evaluate whether the mathematical models simulate the results of the humans are listed below.

- There will be no differences between color-fused, monochrome-fused, infrared, and intensified light image performance predictions by matched filter evaluation and the results of human factors testing applied directly to the same images. If rejected this would support that the matched filter process is not an accurate method for predicting the performance of the HVS.
- There will be no differences between the performance predictions of the Local and Global matched filters for the same images and preprocessing. This should not be true, as the local filter should perform better; however, if no difference is found, then this tends to suggest that the matched filter concept of "tuning" the filter to negate the effects of the background may not have the desired effect.
- The results of the matched filter process on an image will show the same trends as that of an ordinary template matching process.
- There will be no differences between the color-fused image performance predictions by the contrast-based IQM and the results of human factors testing applied directly to the same images. If rejected, this would support that the contrast-based IQM is not an accurate method for predicting the performance of the HVS.

III. MODEL

A. THEORETICAL DEVELOPMENT OF A MATCHED FILTER

A discrete matched filter for object detection can be developed using a spectral formulation that results in a matrix equation that maximizes the signal-to-noise ratio. The algorithm used by Scribner et al. (1993) develops the two-dimensional matched filter along the following theoretical process. Given than the observed image (F_{in}) may consist of background noise (n) alone or a deterministic signal plus background noise ($s+n$), we have the following:

$$F_{in} = s+n, \text{ or } F_{in} = n.$$

Thus, if h_2 were the 2-D matched filter optimized to maximize the signal to noise-power ratio, then the resulting output (F_{out}) of the matched filter may be expressed as the convolution ($\otimes$) of h_2 with F_{in}.

$$F_{out} = h_2 \otimes F_{in}.$$

The signal to noise-power ratio is defined simply as,

$$\frac{S_2}{N_2} = \frac{spectral\ signal}{spectral\ noise\ power},$$

where the 2-D spectral signal (S_2) is defined in terms of the signal in the absence of noise. Since the dimensions of the target are usually smaller than the dimensions of the background, the target is padded with zeros to the dimensions of the background while keeping the target centered in the middle. This padded target is then converted from the spatial domain to the spectral domain through a 2-D fast fourier transform, $fft2(\cdot)$.

$$S = fft2(s).$$

An illustration to help visualize the conversion of a 2-D image from the spatial domain to the spectral domain can be seen in Figure 7, which first shows the spatial representation of two points on the horizontal axis of the spectral domain. These two points are low-frequency points due to their close proximity to the center. The corresponding spatial domain representation is a vertical white band with two dark bands representing a sine-wave corresponding to its low-frequency placement. In the second frame, two more points are added to the vertical axis at the same low-frequency, which produces the equivalent sine-wave "bands," but in the horizontal orientation. As points of higher frequencies are added to the spectral domain plot, a more and more complex "pattern" emerges for the spatial "image."

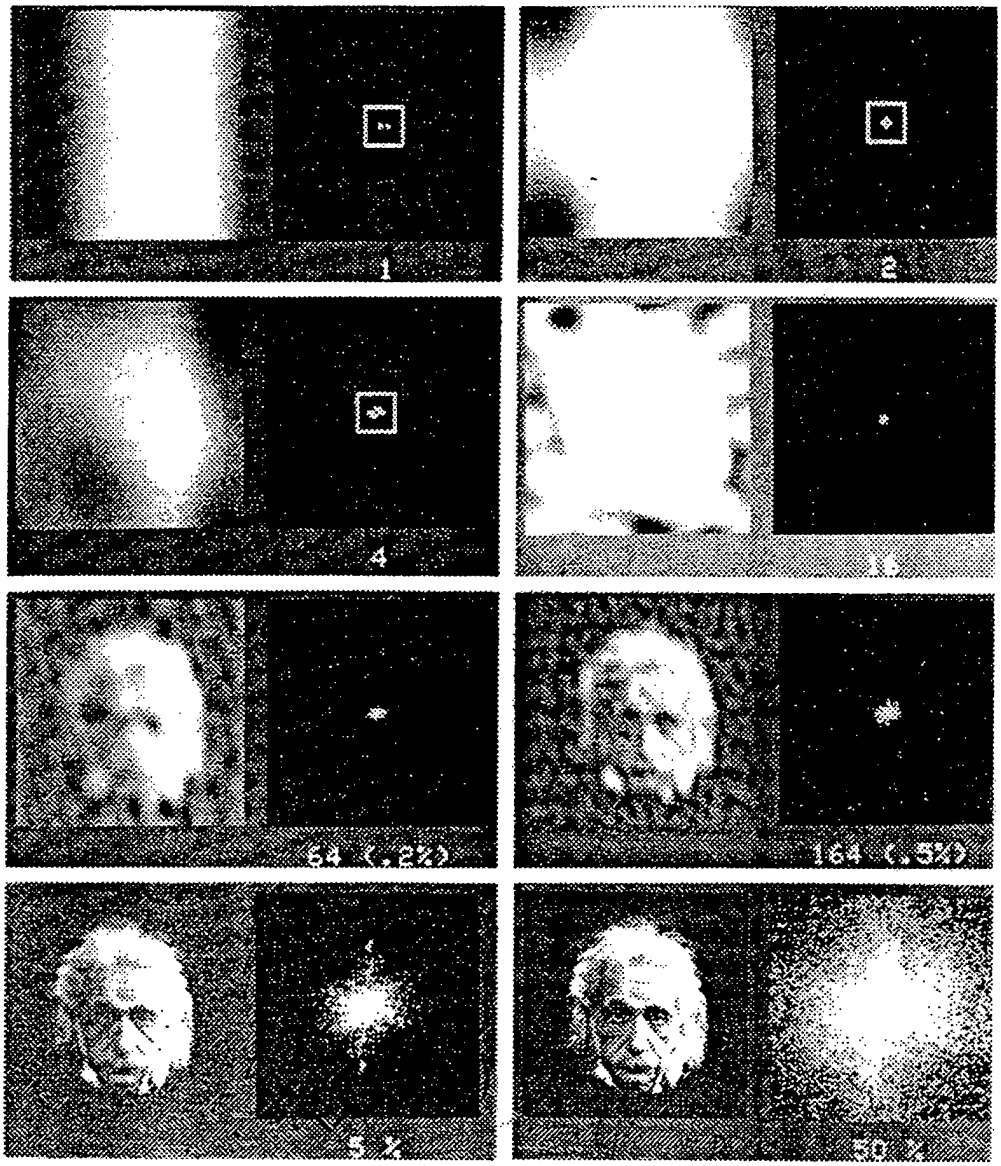


Figure 7. A figure showing the relationship between the 2-D spatial domain image of Einstein and the image's spectral domain representation. The first four frames are shown at a much lower scale than those of the last four frames. This was done so that the individual points could be distinguishable. The numbers below the spectral domain plots represent the number of components represented (first four frames) or the percentage of all the points represented. (De Valois, R. L., and De Valois, K. K, 1988)

Carrying this concept a step further, the spectral "power" of an image is the square of the image's spectral domain representation. The 2-D noise power (N_2) is defined in terms of background noise only and is the spectral power of the noise. Producing the background

noise spectral power is accomplished by converting the noise from the spatial domain into the spectral domain and squaring.

$$Noise\ power\ (N_2) = \{fft2(n)\}^2.$$

Next, the noise power is high-pass filtered in the spatial domain using an *fft* shifted two-dimensional Hanning window. This requires 2-D inverse fast fourier transforming (*ifft*2(·)) the spectral noise power back to the spatial domain and carrying out an element-by-element multiplication of the shifted Hanning window with the spatial noise power. Once this filtering is accomplished, the result is transformed back to the spectral domain. This high-pass filtering "smoothes" the background "energy" by attenuating the background "energy" contained in the lower frequencies. Thus, the background noise used to optimize the filter consists of an average "energy" image with only the "relative energy" contrast of the target on background preserved. Thus, substituting the smoothed noise power (N_s) into the signal to noise-power ratio and transforming back to the spatial domain results in a solution for the matched filter.

$$h_2 = ifft2\left(\frac{conj(S_2)}{N_2}\right).$$

The conjugate on the signal "inverts" the filter in the same way a transpose operation would if done in respect to the minor axis of the matrix. This conjugate is carried out since the filtering process is done in such a way that the corresponding coefficients are 180 degrees out of phase with respect to the minor axis.

With respect to the physical size of the matched filter, a conscious attempt to create a filter larger than the original target was based on the importance of evaluating

some of the surrounding background noise both around the target and within the area in which the target is specifically defined. The amount that the filter was expanded beyond the dimensions of the actual target depended upon the size of the target. Larger targets should be expanded in the same proportions that a smaller one would. Thus, the matched filter developed was expanded in the vertical and horizontal directions by two times the square-root of the vertical and horizontal dimension of the target respectively. For example, in the case of the cylindrical storage tank target having dimensions of 23 pixels in height by 89 pixels in width (23x89), the filter developed would have dimensions of (33x107). This procedure develops the 2-D spatial filters which are used on the single-band or fused monochrome images.

For color-fused images, three-dimensional (3-D) "color" matched filter coefficients are produced, in addition to the 2-D filters, for each color band by performing a three dimensional *fft*, denoted *fft*3(·), on the 2-D targets and backgrounds. For example, if the image were to be fused from an intensified light image and an infrared image, in addition to the creation of the two spatial 2-D filters, two spatial-spectral 3-D filters are also produced for the IR and the IL. Creating the 3-D matched filter coefficients is accomplished using the same theoretical procedure explained above; however, it is carried out on the two bands combined into a 3-D structure. This third dimension is developed by "stacking" the results of the *fft*2 of the individual band targets and backgrounds into a three dimensional matrix and performing a third *fft* with respect to the z-axis. A spectral domain 3-D matched filter (H_3) developed in this manner is expressed,

$$H_3(\vec{k}) = c_0 \frac{S_3^*(\vec{k})}{N_3(\vec{k})} \exp(-i\vec{k}\vec{r}_0),$$

where S_3 is the 3-D signal representation in $\vec{k} = \left[k_x\ k_y\ k_f\right]$, N_3 is the 3-D spectral noise power (Satysher, Scribner, and Kruer, 1997). Thus, the spatial filter is,

$$h_3(x, y, \lambda) = ifft3\left(H_3(\vec{k})\right),$$

which involves carrying out an *ifft* with respect to the z-axis followed by a *ifft*2 resulting in the spectral-spatial matched filters. All of these filters are then stored automatically into a file with a filename consisting of the image name, target name, the processing options specified, and an extension of ".coef." The MATLAB code generating the global and local matched filters is listed in Appendix E.

B. THEORETICAL DEVELOPMENT OF THE CONTRAST-BASED IMAGE QUALITY METRIC

The contrast sensitivity function (CSF) represents the transform function between the image being viewed and the spatial frequency response of the HVS. Based upon the averaged results of several contrast threshold experiments, parameter estimates of the actual CSF have been obtained. Various mathematical models have been proposed based on the assumption that this CSF is a smooth curve. The generalized model of human contrast sensitivity developed by Pelli, Rubin, and Legge proposed a parabolic fit to this CSF. Pelli et al. (1990) offered this simple non-linear parabolic fit as a satisfactorily accurate, easily implemented, and fast model.

A Visual C++ program developed by Dr. Sanjoy Das implements the general Pelli model to compensate for the spatial display degradation of a CRT by weighting the intermediate frequencies of a digitized image. This is accomplished by transforming an image from the spatial domain to the spectral domain by means of the *fft2*, and then radially weighting those frequencies associated with the experimentally estimated CSF according to Pelli's non-linear parabolic model. This reweighted image is then transformed back into the spatial domain by means of the *ifft2*. The resulting image is now weighted in accordance with the proposed Pelli CSF model. This program performs this reweighting on single-band and monochrome images as is. All pseudo-colored images must first be split into their RGB components and each color band reweighted individually. The Visual C++ code developed by Dr. Sanjoy Das to accomplish the above is shown in Appendices L and M.

C. SETUP OF PLANNED COMPARISONS BASED UPON THE HYPOTHESES

Instead of analyzing the data to see if there is one or more overall experimental effects, the planned comparisons technique will be used to answer a number of individual questions posed at the outset of the experiment. These proposed hypotheses around which this experiment was designed are examples of specific questions to be answered separately from the experimental results. The planned comparisons sensitivity "model" constructed to evaluate these proposed hypotheses have the form: A x B x C x D. The four different filters evaluated in this experiment are represented in group "A": the global

matched filter (G), the local matched filter (L), the template matching filter (T), and the "Human" filters (H). The "Human" filters consists of the five subjects who participated in the human factors experiment (Sampson, 1996). The different image scenes used in this experiment are represented in group "B": a scene containing a "rectangular" target or building ("rct"), a scene containing a "cylindrical storage tank" ("tnk"), and a scene containing an airfield "tower" ("twr"). The four different sensor "classes" used for each of the scenes in this experiment are represented in group "C: infrared (IR), intensified light (IL), pseudo-color fused (F1), and monochrome fused (F2). Lastly, the three different target positions used for each scene/sensor combination are represented in group "D: position "1" is the original target position within the scene; positions "2" and "3" were derived by digitally moving the target to two other locations within the corresponding scene.

D. MODELING A NORMAL DISTRIBUTION FOR CENSORED REGRESSION

The human factors data measuring the amount of time to located the target or to determine its absence within an image can be modeled with a Normal distribution with a mean equal to 706.3 μsecs and a standard deviation of 128.7 ($\underline{ks}$=0.1402, $\underline{p}$=0.0712).

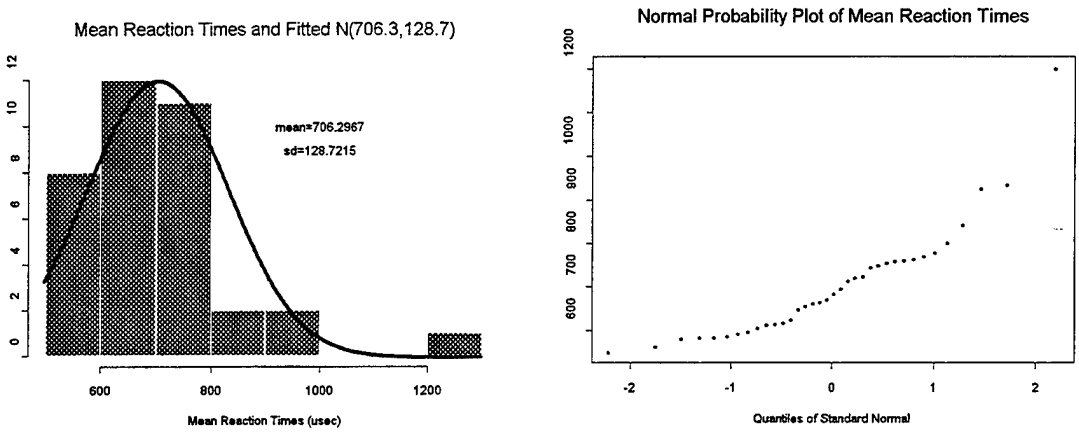


Figure 8. The mean reaction times from the human factors experiment (Sampson, 1996) fitted with a N(706.3, 128.7).

The censored normal function can be used in cases in which the dependent variable is censored to a maximum value. Censoring of this type is known as censoring from "above." This type of data requires a mixed distribution with the continuous Normal portion existing as usual for values less than the "censor" value. At the "censor" value, the Normal distribution "jumps" to a discrete probability representing the Normal probabilities above the "censor" value. This kind of distribution exists for the sensitivity data calculated from the human factors experiment (Sampson, 1996) and will be applied to the sensitivities of the mathematical models being evaluated. For the purposes of this experiment, a maximum sensitivity of 4.6527 was imposed.

This censored regression technique applies because ordinary least squares regression becomes inconsistent as the percentage of censored values within a dataset increases. Unfortunately, there are no hard and fast rules dictating what percentage of the dataset is allowed to be censored without affecting the ordinary least squares (OLS)

estimates. It is generally agreed that if less than ten percent of the dataset is censored, then the effects of censoring on the OLS is minimal. It is also generally agreed that somewhere around forty percent of the dataset being censored will most likely result in severe effects. The safest route is to address the censoring issue directly by producing the maximum likelihood estimates for the coefficients of the censored dependent variable and comparing it to those estimated by OLS and interpreting the severity.

The regression coefficients for a censored Normal distribution can be obtained with the log-likelihood estimation providing the maximum likelihood estimators for the coefficients of a censored regression.(Greene, 1997).

$$\ln L = \sum_{\forall y_i < a} \frac{-1}{2}\left[\ln(2\pi) + \ln(\sigma^2) + \frac{(y_i - \vec{\beta}'\vec{x}_i)^2}{\sigma^2}\right] + \sum_{\forall y_i = a} \ln\left[\Phi\left(\frac{a - \vec{\beta}'\vec{x}_i}{\sigma}\right)\right].$$

The $\Phi(\cdot)$ notation represents the normal CDF and "a" represents the censor value. The above equation is a sum of two parts. The first part sums the classical regression for the non-limit observations, followed by a sum of the relevant probabilities for the "censored" observations. Using Olsen's (1978) reparameterization of $\vec{\gamma} = \vec{\beta}/\sigma$ and $\theta = 1/\sigma$, the model simplifies tremendously, allowing for straightforward calculation of the Hessian. The estimation then lends itself to Newton's method.

$$\ln L = \frac{n}{2}\left(\ln(2\pi) + n\ln(\theta)\right) - \frac{1}{2}\sum_{\forall y_i < a}\left(\theta y_i - \vec{\gamma}\vec{x}_i\right)^2 + \sum_{\forall y_i = a}\ln\left(\Phi(a\theta - \vec{\gamma}\vec{x}_i)\right).$$

Letting, $\alpha_i = a\theta - \vec{\gamma}\vec{x}_i$, $\varepsilon_i = \theta y_i - \vec{\gamma}\vec{x}_i$, $\lambda_i = \dfrac{\phi(\alpha_i)}{\Phi(\alpha_i)}$, and $\zeta_i = \lambda_i\alpha_i - \lambda_i^2$,

results in the elements required for $\nabla f(\vec{p})$ and $\nabla^2 f(\vec{p})$, where,

$$\vec{p} = \begin{bmatrix} \vec{\gamma} \\ \theta \end{bmatrix}.$$

The elements required for $\nabla f(\vec{p})$ are,

$$\frac{\partial \ln L}{\partial \vec{\gamma}} = \sum_{\forall y_i < a} \varepsilon_i \vec{x}_i - \sum_{\forall y_i = a} \lambda_i \vec{x}_i \text{, and } \frac{\partial \ln L}{\partial \theta} = \frac{n^2}{2\theta} - \sum_{\forall y_i < a} \varepsilon_i y_i + a \sum_{\forall y_i = a} \lambda_i .$$

The elements required for $\nabla^2 f(\vec{p})$ are,

$$\frac{\partial^2 \ln L}{\partial \vec{\gamma}^2} = \sum_{\forall y_i < a} \left(-\vec{x}_i \vec{x}_i\right) - \sum_{\forall y_i = a} \zeta_i \vec{x}_i \vec{x}_i \text{, } \frac{\partial^2 \ln L}{\partial \theta \vec{\gamma}} = \sum_{\forall y_i < a} y_i \vec{x}_i - a \sum_{\forall y_i = a} \vec{x}_i \zeta_i \text{, and,}$$

$$\frac{\partial^2 \ln L}{\partial \theta^2} = -\frac{n^2}{2\theta^2} - \sum_{\forall y_i < a} y_i^2 + a^2 \sum_{\forall y_i = a} \zeta_i .$$

Once the maximizing parameters ($\vec{p}$) have been determined, the coefficients and standard deviation can be recovered since $\vec{\beta} = \vec{\gamma}/\theta$ and $\sigma = 1/\theta$. In addition, an estimated asymptotic covariance matrix is determined using a J matrix and the Hessian for $\vec{p}$ with,

$$J = \begin{bmatrix} \frac{\partial \vec{\beta}}{\partial \vec{\gamma}'} & \frac{\partial \vec{\beta}}{\partial \theta} \\ \frac{\partial \sigma}{\partial \vec{\gamma}'} & \frac{\partial \sigma}{\partial \theta} \end{bmatrix} = \begin{bmatrix} \frac{1}{\theta} \vec{I} & \frac{-1}{\theta^2} \\ \vec{0}' & \frac{-1}{\theta^2} \end{bmatrix},$$

where $\vec{I}$ is an identity matrix and $\vec{0}'$ is a vector of zeros.

The resulting covariance matrix is then equal to,

$$COV = J\left[-\nabla^2 f(\vec{p})\right]^{-1} J' .$$

IV. METHODS

A. MATCHED FILTER PROGRAM DEVELOPMENT

The experiment constructed for this thesis was to use a matched filter program as well as an IQM program to quantify the expected performance of various infrared, intensified light, fused monochrome, and pseudo-color fused images in order to compare them to the results of human target detection performances.

The matched filter program was based upon the approach used by Scribner et al. (1993) to evaluate the images used in the human factors experiment (Sampson, 1996). In order to remain consistent with the human factors experiment, the matched filter program used the same targets contained within the human factors images rather than the synthetic targets used by Scribner et al. (1993). Also, the images were not aerial photographs of air fields and other similar areas of interest taken from a relatively high altitude. Sampson's images were taken at low altitude and contain a horizon separating earth and sky. Thus, the program included an option to develop a matched filter based only on the area in which the target could sensibly exist, resulting in an option to create a global or local matched filter based upon the background of interest. This was accomplished by allowing specification of a rectangular area (termed the specified field) by inputting the coordinates of the upper-left (Xmin, Ymin) and lower-right (Xmax,Ymax) corners in which the target's center-of-mass can be placed. A third filter option, evaluation by way of template matching, was also developed for comparison to the matched filter evaluations.

The MATLAB 5.0 program code for the matched filter appear in Appendices A through J and is divided into the major sub-sections of the program. Appendix A contains

the main m-file code, which coordinates calls to other m-file units. These other units accomplish: getting data, processing options, and carrying out any preprocessing (see Appendices B through D); deriving the specified 2-D and 3-D matched filter coefficients (see Appendices E and F); filtering the images with the developed filters (see Appendices G and H); and displaying the results (see Appendices I and J).

B. BACKGROUND AND TARGET IMAGERY

Three nighttime scenes were digitally imaged using an early prototype fusion sensor system developed by Texas Instruments and the Night Vision Electronic Sensor Directorate (Palmer, Ryan, Tinkler, and Creswick, 1993). These images were collected from a low-light visible image intensifier (.6-.9μm) and a first generation forward-looking infrared (8-12μm) mounted on a UH-1N helicopter. This system was designed to simultaneously record digitized IR and IL nighttime images. Images were collected during starlight (10^{-3} lux) to full moon (10^{-1} lux) conditions. These IR and IL bands were then used to produce the monochrome fused and the pseudo-colored fused images resulting in a total of four sensor "classes" for each of the three nighttime scenes. The images and targets used for this experiment were identical to those used in human factors assessment of fused images on observers' reaction time in target detection (Sampson, 1996). These images were all saved in the tagged index file format (TIFF) with image registration already accomplished.

The first step in developing the matched filters was to separate the targets from the background. This was accomplished using Adobe Photoshop 3.0 on an Apple Power

Macintosh 6100/66 Power PC. The rectangle used to crop the target from the image was kept as small as possible. Additionally, in order to ensure that the matched filter developed from these targets would have a single center-point (correlating to the matched filter output), the vertical and horizontal dimensions of the rectangle were an odd number of pixels. Lastly, each target from a particular scene had to have the same target dimensions since the 3-D filters are developed from the respective targets of the bands being fused.

Once the targets were cropped from each of the four image "classes" of a scene, each was enlarged, and any background around the target was "erased" by hand drawing a black outline around the target and painting everything not part of the target black (corresponding to a grey-level equal to zero). This resulted in the grey-level pixels of the target existing in a field of zeros, thus providing the pure target signal in the absence of noise.

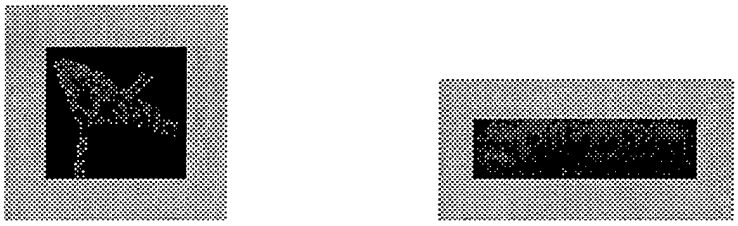


Figure 9. Example of the satellite dish target and the cylindrical tank target with the background noise removed.

This was saved as the target in TIFF format, and then a target "template" was developed and saved. In the template, the background area around the target is white (corresponding to a grey-level equal to one), and where the target was located is all black (zeros).

Figure 10. Example of the templates used with the satellite dish target and the cylindrical tank target.

This template is used to "cut out" the area in which the target is to be applied. Thus, since everywhere around the target is a value of one, an element level multiplication of a background segment with the template preserves the grey-levels multiplied by one. The area in which the target is to be placed is all zeros, which removes the area of the background corresponding to the shape of the target. This allows the target to be added to the background efficiently through matrix addition.

C. FILTERING AN IMAGE WITH A MATCHED FILTER

The background scene (area to be filtered) and specified field (area in which the target's center of mass can exist) are directly related since the background scene is always physically larger than the specified field. The background scene can be as large as the original image itself, or as small as the dimensions of the filter being used. On the other hand, the specified field has a maximum size limited to the "central" portion of original image and can be as small as a single point. This "central" portion excludes the outer edges of the background. When the (Xmin,Ymin) and (Xmax,Ymax) coordinates of the specified field are input, the valid x-coordinate range is limited to those coordinates that are located greater than half the width of the filter being used from the left or right edges of the image. Similarly, the valid y-coordinate range is limited to those coordinates that

are greater than half the height of the filter being used from the top or bottom edges of the image. This is due to the specified field being the area in which the target's center point can be located. By restricting the range limits in this manner, placing the filter center anywhere within the specified field results in the entire filter remaining on the background scene. This prevents any questions of validity regarding the filter's output since no convolutions require padding with zeros. Thus, once the specified field has been input, the background scene to be filtered is then expanded outward from the coordinates of the specified region by half the width and height of the filter in the respective directions.

When filtering a single color image, such as an infrared or a monochrome fused image, only the corresponding 2-D spatial filter is used. Optimally, the filter would be evaluated twice at every point within the specified field of the image--once to calculate the matched filter's output value when applied to the background noise (N) only, and a second time to calculated the matched filter's output when the target has been applied to that same location (signal-plus-background-noise (SN)). The scene in Figure 11 shows the fused monochrome "cylindrical tank" scene without the target present.

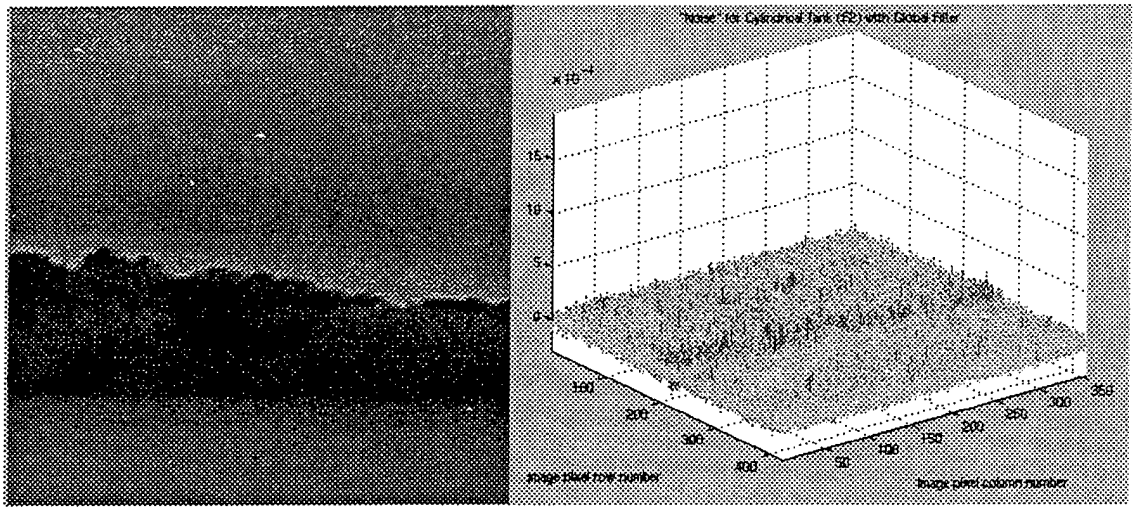


Figure 11. The cylindrical tank scene on the left does not contain the target. On the right are the 2-D Global Matched Filter output values evaluated at every pixel within the scene (Courtesy of the U.S. Army Night Vision and Electronic Sensors Directorate).

Evaluating the matched filter's output for N only can be accomplished very efficiently by convoluting the entire background scene (without the target present) with the 2-D matched filter and considering only the "central" portion of the output. Thus by performing a single convolution, we have the results of evaluating that filter at every valid point within the background noise only. The plot in Figure 11 shows the 2-D global matched filter output values for each pixel of this "background noise" image. In this case, the filter is positively correlated, and the higher the filter output value, the more that area of background noise "matches" the target.

In order to evaluate the filter value in the SN case, the filter must be convoluted with a background in which the target is present, as in the scene depicted in Figure 12. The plot in Figure 12 also shows the same 2-D global matched filter when convoluted

with the same background but the target present. The large "spike" correlates to the pixel at (344,359), the location of the target's center-of-mass.

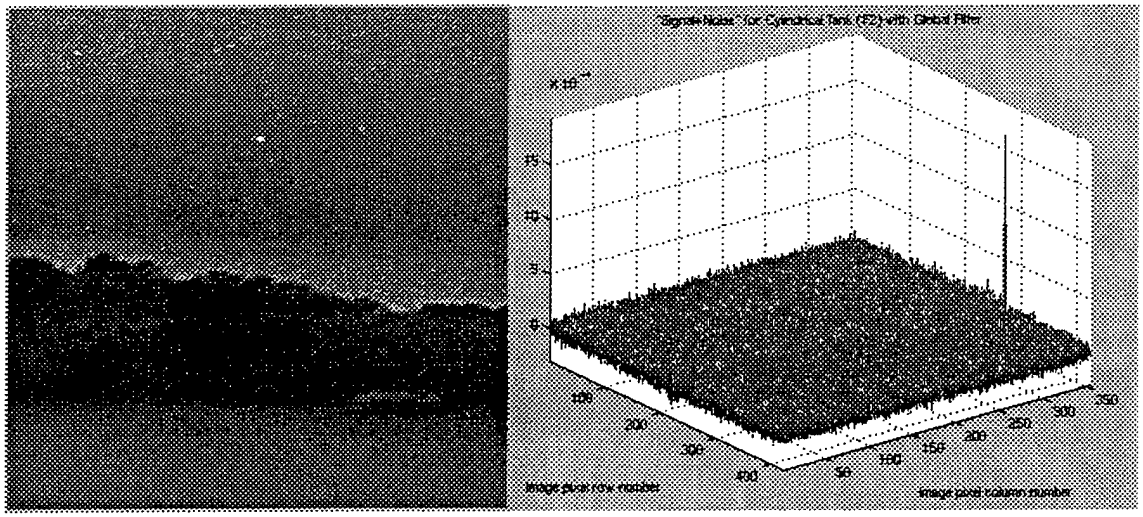


Figure 12. The cylindrical tank scene on the left with the target. On the right are the 2-D Global Matched Filter output values evaluated at every pixel within the scene. The "spike" correlates to the target's location (Courtesy of the U.S. Army Night Vision and Electronic Sensors Directorate).

This single SN value at (334,359), along with the single N value (from pixel (334,359) of the filter's N output values), make up a single "evaluation" pair corresponding to the point (334,359). Consequently, evaluating the background noise with the target inserted at every point could require tens of thousands of iterations of inserting the target centered at the point being evaluated, convoluting this with the respective filter, storing the filter's output value at that single point, and repeating for every pixel within the specified region. Since the images used for these evaluations are 456 pixels by 460 pixels (456x460) with various filters (largest being (65x69) pixels), attempting to evaluate such a large number of points would be extremely time-consuming. Thus, a sub-optimal approach was performed which allowed a step size to be specified for both the horizontal and vertical directions. The vertical and horizontal step size used for these evaluations roughly

correlated to one-fifth of the vertical and horizontal dimensions of the target being used. Figure 13 shows the suboptimal N and SN filter output values for the same scenes above, in which the specified region was limited to the grass field with a step size of 18 pixels horizontally and 4 pixels vertically (approximately one-fifth the target's dimensions of (23x89)).

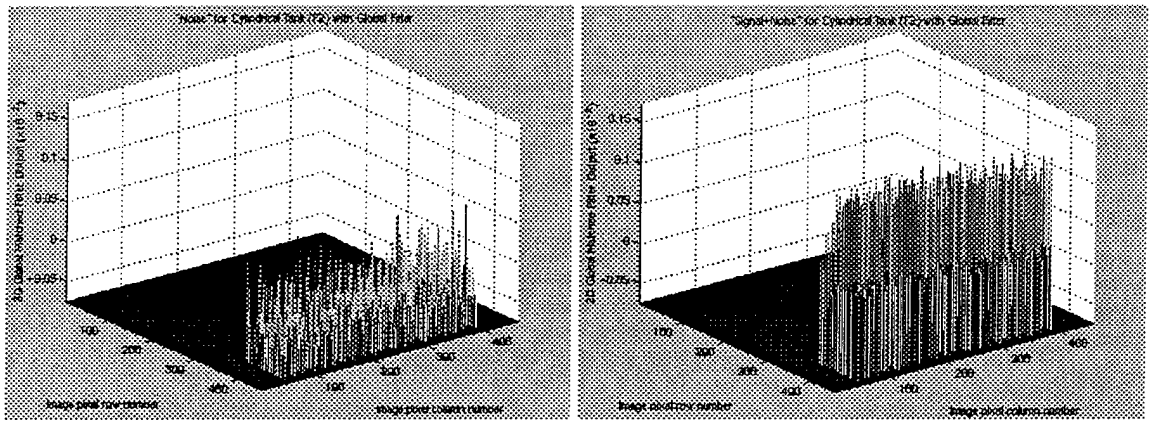


Figure 13. Shown above are the 2-D global matched filter output values for a suboptimally sampled specified field corresponding to the grass field portion of the "cylindrical tank" scene. On the left are the results of evaluating the background noise only. On the right are the results of evaluating the signal+noise at the corresponding coordinates.

In the case of evaluating the performance of more than one "color" band, the individual bands are evaluated as described above for both background noise and signal-plus-background-noise followed by the respective 3-D spatial-spectral filters. The 3-D filter output values for N are then summed into a single 3-D N output value (similarly for SN). All of the 2-D SN and N output values and the 3-D SN and N output values are then stored to the hard drive using the same filename convention described for saving the matched filter coefficients, except with a ".dat" extension. The MATLAB code developed to perform this filtering procedure is listed in Appendix G.

D. INTERPRETING MATCHED FILTER OUTPUT VALUES

A matched filter is a spatial filter that provides a measure of the correlation between the background scene and the target (Pratt, 1991). Thus, the calculated matched filters can be thought of as acting much like a monotonic function, with the filter output values for N having a theoretical mean of zero due to optimizing the filter to reduce the effects of the background noise. In contrast, with the matched filter optimized for the signal to noise-power ratio, the filter output values for SN have a theoretical mean other than zero. As a result, the mean of the SN filter output distribution could end up being greater than (a positive correlation) or less than (a negative correlation) the mean of the N filter output distribution. Either way, a completely negative or a completely positive correlation results in "discriminating" the target from the background noise, which has relatively little correlation. In signal detection theory, the convention normally used to illustrate the relationship between the N and SN distributions graphically depicts the mean of the SN distribution to the right of the N distribution's means. Thus, as the signal's strength becomes weaker, the mean of the SN distribution shifts to the left until finally, there is no difference between the N and SN distributions (signal too weak to distinguish).

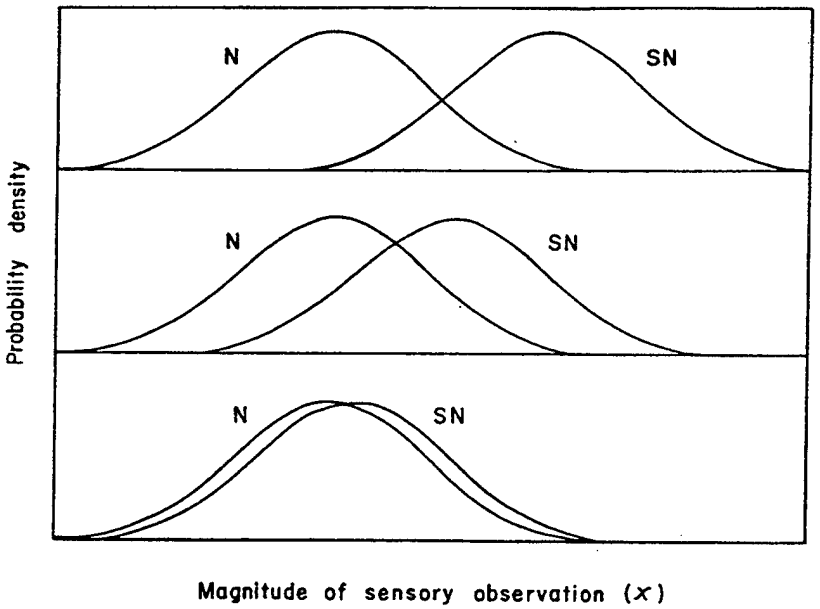


Figure 14. Depicts the N and SN distributions as the signal's strength becomes weaker until, finally, there is no difference between the N and SN distributions (signal too weak to distinguish) (Gescheider, 1985).

In the same way, the increasing and decreasing distance between the means of the N and SN distributions as the target strength increases or decreases respectively is preserved by the matched filter, whether the correlation is positive or negative. The reason this negative correlation, or "inversion," occurs has to do with matching the background of the target to a smoothed average "energy." When the target is inserted into a background of lower or higher "energy" and filtered, the resulting contrast leads to either a negative or positive filter output value (Scribner, personal communication). However, in order to standardize the matched filter output results to conform to the accepted conventions of signal detection theory, those cases in which a negative correlation is determined to exist, all of the matched filter output values are multiplied by -1, resulting in the more conventional distribution relationship. This negative correlation exists when the mean of the matched filter SN output values is less than the mean of the matched filter's N output

values. Figure 15 illustrates the histograms of the resulting 2-D Global matched filter N and SN output values for the "Cylindrical Tank" monochrome-fused image.

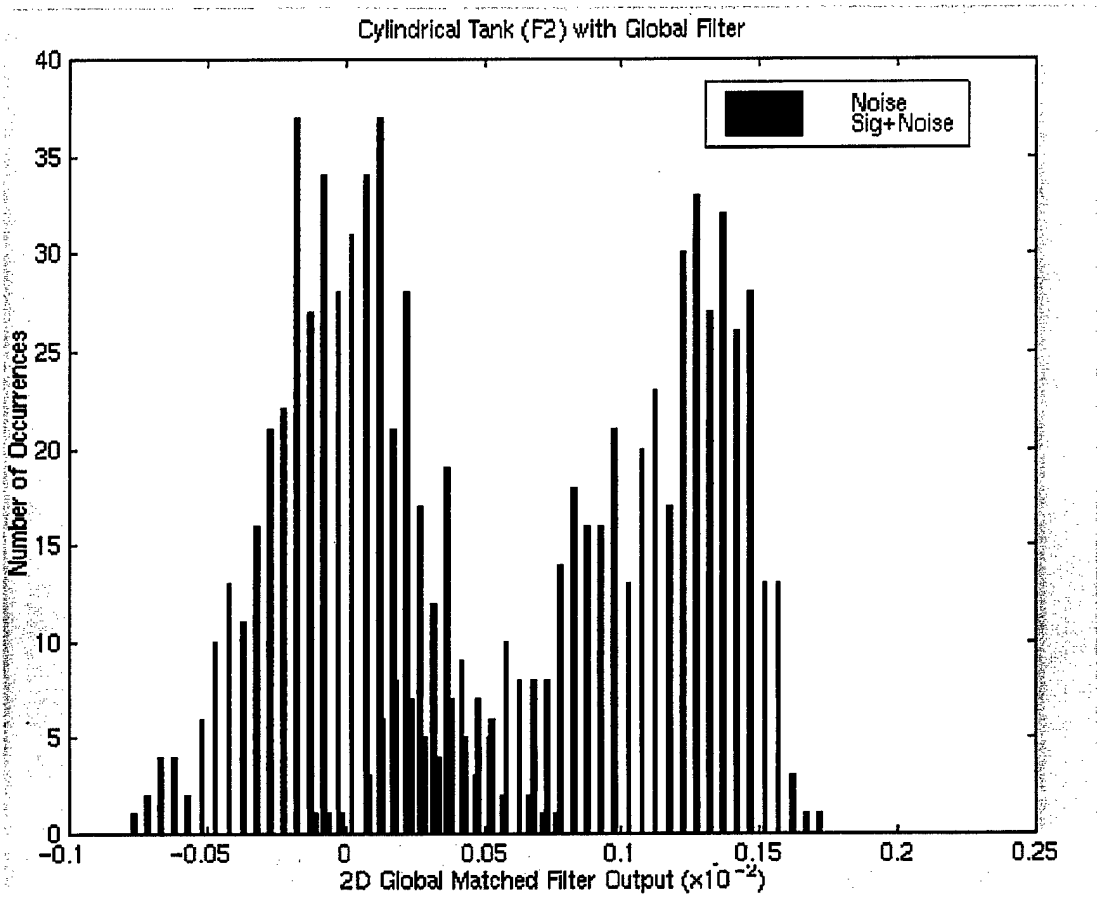


Figure 15. The estimated N and SN distributions for the 2-D global matched filter used on the Cylindrical Tank (F2) scene obtained through suboptimal sampling.

The specified horizontal and vertical step size used to filter each image resulted in approximately 400 separate point evaluations within the specified field. Thus, Figure 15 represents an empirical estimate of the N and SN distributions for the filter on a particular scene/sensor combination. The MATLAB code used to graphically display the results of the filter data is listed in Appendix J.

E. DEVELOPMENT OF EMPIRICAL RECEIVER-OPERATOR CURVES

A single filter output value taken by itself has no interpretive value. However, in relation to other matched filter outputs for N and SN from the same image, it becomes possible to predict both "hit" rates and "false-alarm" rates within the field evaluated by the matched filter. This can be accomplished by varying a threshold criterion (critical point) through the range of filter output values. Any SN filter value greater than this threshold criterion will be designated as a successful detection or "hit", while any N filter value greater than this threshold criterion will be designated as a false detection or "false-alarm." The total number of "hits" and "false-alarms" divided by the total number of trials (points evaluated) produces the "hit" rate (Pd) and the "false-alarm" rate (Pfa).

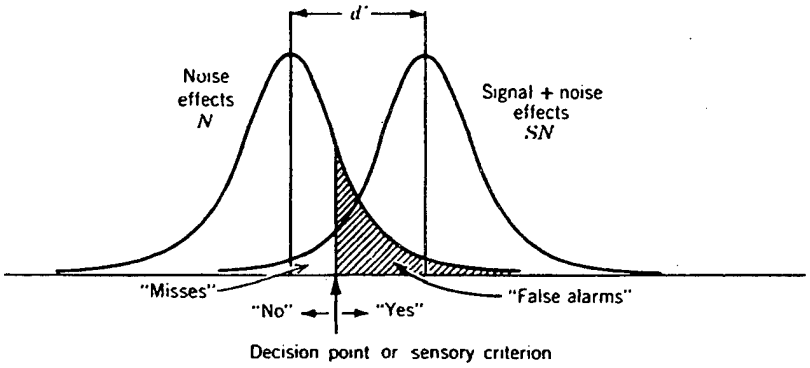


Figure 16. This illustrates the relationship between the threshold criterion (critical point) and the rate of detection (Pd) and rate of false alarm (Pfa) for a single threshold criterion (Schiffman, 1990).

Thus, if we define the criterion as the maximum SN value, then there are no "hits" or "false-alarms" since there are no values greater than the criterion (the maximum). As the criterion is lowered, the number of "hits" begins to increase and the Pd becomes greater than zero. However, the "false-alarm" rate will also begin to increase as the threshold criterion starts to enter the range of the N distribution. As the criterion reaches the

minimum of the N filter output values, a hit rate of 100 percent would have already been achieved. However, since all of the N filter output values are greater than this minimum criterion, there is also a 100-percent false-alarm rate. Thus, an empirical ROC can be plotted simply by varying the threshold criterion from the maximum SN filter output value down to the minimum N filter output value, determining the hit and false-alarm rates at each threshold criterion, and plotting the resulting (Pd, Pfa) pairs. These ROC plots characteristically start at the (Pd=0%, Pfa=0%) point (threshold criterion at the maximum filter value) and end at the (Pd=100%, Pfa=100%) point (threshold criterion at the minimum filter value).

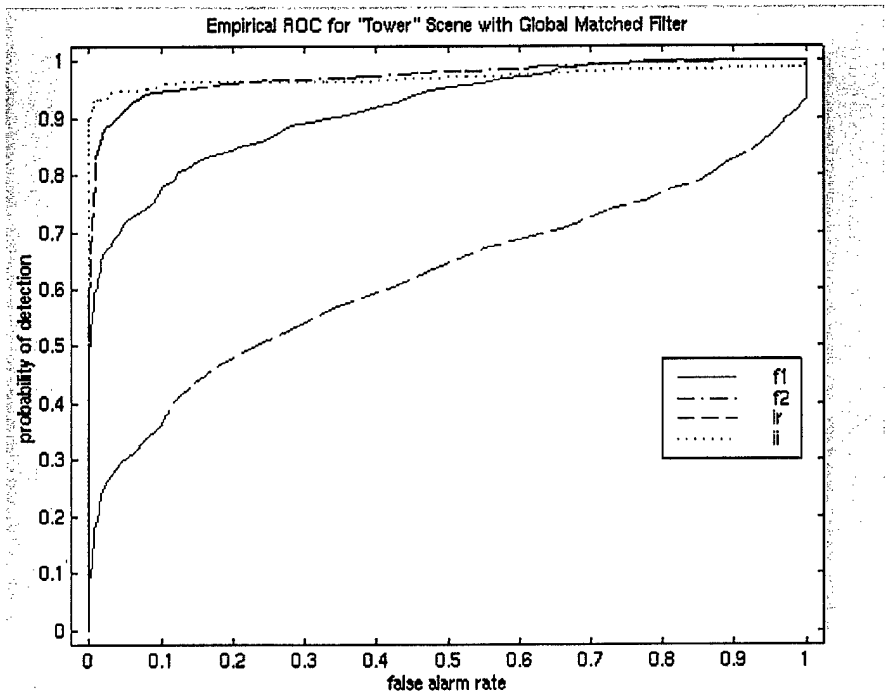


Figure 17. Empirically-derived ROC plots for the 2-D global matched filters (including sensors IL, IR, and F2), as well as for the 3-D global matched filter (F1).

The MATLAB 5.0 code to threshold and produce the empirical ROC plot is listed in Appendix I for the data placed in the "summary" variable. A separate m-file code was

developed to allow the recall of any of the ROC plots for later reference or analysis. This code is listed in Appendix J.

F. DETERMINATION OF THE HUMAN VISUAL SYSTEM SENSITIVITY AND BIAS

The subjects of the human factors experiments (Sampson, 1996) participated in a forced response (yes or no) two stimulus experiment in which one of the sensor "classes" from one of the three scenes were temporarily shown with or without a target present, and the subject's reaction time for each response was recorded. However, in addition to recording the reaction time, the observer's accuracy in response and the position of the target (if present) were also recorded. Thus, if the temporarily shown image contained a target (condition A), and the observer responded that a target was present (condition A), then a valid "detection" was recorded. But if the temporarily shown image did not contain a target (condition B), and the observer responded that a target was present (condition A), then a "false alarm" was recorded. For his experiment, Capt. Sampson developed images in which the target's location had been digitally "moved" among three positions. Thus, the overall Pd and Pfa can be calculated for each of the three positions for any sensor "class" of any particular scene. This was accomplished by taking the raw data results of the human factors experiment (Sampson, 1996) and extracting the columns pertaining to: (1) the name of the person observing this image, (2) which sensor "class" of the scene was shown, (3) which scene was shown, (4) the position of the target (or no target), (5) the target condition presented, and (6) the observer's response. This

produced a data set consisting of these six columns and 3240 rows. These data are the complied results of the five subjects whose responses were considered unbiased.

An S-plus function called "CalcSensAndBias" (listed in Appendix K) was created to accept a dataframe with these six columns and any number of rows. In order for this function to execute correctly, the information must be formatted in the same column order as described above. This function calculates the number of correct detections ("AA" pairs) and false alarms ("BA" pairs) for each person for every scene/sensor combination for each target position. For the overall scores, the "AA" pairs of each subject were summed for each scene/sensor combination (similarly for the overall "BA"'s). From these, we calculate the Pd's and Pfa's by dividing by the number of observations for that particular case. As discussed in Chapter 2, solving for the sensitivity (d') is simply the inverse of the normal distribution for Pd minus the inverse of the normal distribution for Pfa. The sensitivity value can then be related to a general ROC plot since many values of Pd and Pfa can result in the same sensitivity.

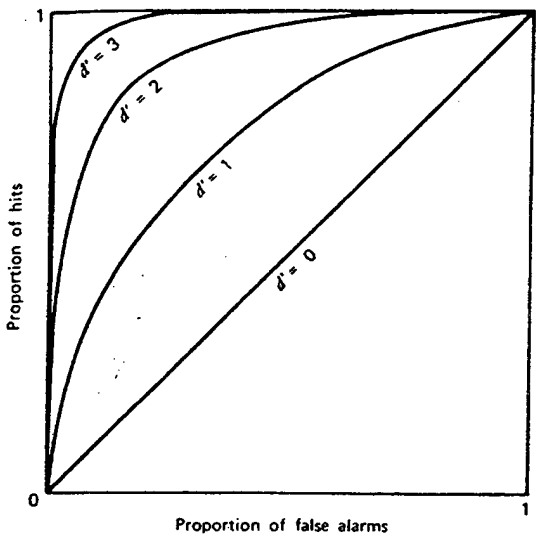


Figure 18. Examples of the general ROC plots for d'= 0, 1, 2, and 3. (Schiffman, 1990)

Furthermore, calculating the bias associated with the Pd and Pfa pairs is just as straightforward. Also discussed in Chapter 2, bias is simply the pdf of the normal distribution for Pd divided by the pdf of the normal distribution for Pfa. The computed sensitivities and bias are listed the Chapter 5 and in Appendix P for each subject (as well as for overall) across the combinations of scene/sensors for the three different positions.

G. DETERMINATION OF MATCHED FILTER SENSITIVITY AND BIAS

In evaluating the sensitivity of the human observers, the targets were placed at three different locations within each image. Thus, the resulting sensitivities calculated for each subject are a measure of the subject's ability to discriminate the target from the background at these locations. Similarly, the ability for the matched filter to discriminate the target from the background at these location is directly related to the matched filter's N and SN output values at these locations. As previously seen in Figure 15, the N and SN empirical distributions reflect how well the global 2-D matched filter can distinguish the target from the background as a whole throughout the field of interest. Thus, the matched filter's sensitivity and bias at a specified location can be calculated by thresholding with respect to how well the same global 2-D matched filter can distinguish the target at the specified location. By using the matched filter's N and SN output values at the specified location as the respective threshold criteria, denoted Nt and SNt respectively, one can estimate the detection rate (Pd) and the false alarm rate (Pfa) existing at this specified location. Any of the global 2-D matched filter SN output values from the specified field which are greater than the Nt value represents a target which is

successfully discriminated from the background noise at that location. However, any of the global 2-D matched filter N output values from the specified field which are greater than or equal to the SNt value represents mistaking the background noise as a target. The Pd and Pfa for the specified location is then determined by counting the number of successful discriminations (hits) and the number of false detections (false alarms) and then dividing by the number of trials. Once the Pd and Pfa have been determined for the specified location, solving for the matched filter's sensitivity and bias for that specified location is merely substitution into the respective equations.

In order to obtain the Nt and SNt values at the locations corresponding to the positions used in the different scenes, an option to filter at a specified point instead of in a rectangular area was created. This uses the same code to filter a specified area, but is limited to a single iteration for the point specified and does not create matched filters. Instead, it uses the filters which have already been developed for the specified scene/sensor/and processing options saved in the corresponding ".coef" file. This ensures that the point is evaluated using the same filter with the same processing options as specified originally when filtering the corresponding specified field. This single point result is then saved with the save name convention as before with a ".dat" extension; however, instead of an "F" for the field option, there is a "P" for the point option. Since the target needs to be evaluated at three different locations per scene/sensor combination, the "P" series goes from "1" to "3", resulting in three ("P1", "P2", and "P3") point data files.

Once the field and associated point data files have been created, the matched filter output values for the point are used as the threshold criteria, and the sensitivity and bias associated with that point location are calculated and displayed. The m-file code to calculate the sensitivity and bias from the stored field and point ".dat" files is listed in Appendix J.

H. OBTAINING THE CSF BASED IQM OUTPUT VALUES

The Visual C++ programs for the CSF-based IQM were compiled and run on a Pentium micro-computer from a DOS window in Microsoft Windows '95. The program to reweight an image according to the parabolic model proposed by Pelli et al. (1990) required two filename inputs. The first was the filename of the image to be reweighted, and the second was the filename with which the reweighted image would be saved as. The image format required for this program was either a PNM or a PGM file format. Thus, the TIFF images were converted to PNM format images using Image Alchemy v1.9.1. Pseudo-colored images required the additional step of splitting the image into its RGB components through the use of a support program called "splitRGB."

The measure of contrast used for quantifying the "reweighted" contrast content within an image was a (9x9) mean-squared contrast filter, which measures contrast as the square of the mean differences summed over a (9x9) area of the image being evaluated. The contrast filter is filtered throughout the entire image and the output results summed to produce the overall contrast content contained within the image.

I. SETUP OF PLANNED COMPARISONS BASED UPON THE PROPOSED HYPOTHESES

Multiple comparisons may be carried out on S-plus 4.0; thus, the contrast model of A x B x C x D was required. The filter contrasts defined were: "cf1", a comparison between the sensitivities resulting from the HVS ("H") and the sensitivities of the global ("G") and local ("L") matched filters; "cf2", a comparison between the sensitivities produced by the global and local matched filters; and "cf3", a comparison between the sensitivities of the local and global matched filters to the sensitivities of the template matching filter.

Contrasts defined for the sensors were: "cs1", a comparison between the sensitivities produced by the pseudo-color fused sensor and the sensitivities resulting from the individual IL and IR sensors; "cs2", a comparison between the sensitivities produced by the fused monochrome sensor and the sensitivities of the other sensors; and "cs3", a comparison between the sensitivities produced by the IR sensor and the sensitivities produced by the IL sensor.

Then, the interactions of interest were defined to include three of the two-way interactions between filter, scene, sensor, and position, as well as two three-way interactions. The S-plus code to accomplish these tasks are listed below:

```
levels(Filter)= "G" "H" "L" "T"
levels(Sensor)= "f1" "f2" "ii" "ir"

cfilt<-cbind(cf1=c( -1, 2, -1, 0), cf2=c( 1, 0, -1, 0), cf3=c( 1, 0, 1, -2))
csens<-cbind(cs1=c( 2, 0, -1, -1), cs2=c( -1, 3, -1, -1), cs3=c( 0, 0, -1, 1))

 mf<- aov(formula = Sensitivity ~ Sensor + Scene + Position + Filter +
       Sensor:Scene +Scene:Filter + Position:Filter + Sensor:Scene:Position +
       Sensor:Scene:Filter, data = D2.data, contrasts= list("Filter=cfilt",
       "Sensor=csens", "Scene=contr.treatment(3)","Position=contr.treatment
       (3)"),na.action=na.omit)
```

J. TREND ANALYSIS OF THE MODELED RESULTS USING CENSORED REGRESSION

Once the associated model output values had been obtained, comparison of the results of the contrast-based IQM and the calculated sensitivities were done using a non-linear censored regression model assuming a normal distribution. A MATLAB function called "CensoredRegrssnMLEs.m" was used to carry out Newton's method using an accepted tolerance of 1×10^{-10}. This function outputs the resulting estimated coefficients maximizing the log-likelihood equation discussed in the Chapter 3 (Models), and the asymptotic covariance matrix. In addition, the associated intermediate calculations are also displayed as the error term iterates toward zero.

The human factors experiment (Sampson, 1996) was conducted such that each subject had an ample amount of time to visually search the image prior to responding. The mean reaction times and the accuracy data collected from that experiment indicated a weak, if any, speed-accuracy tradeoff. This tradeoff normally exists in speeded performance tasks and describes the reciprocity between latency and the number of incorrect responses made. Since this is not the case with the human factors experiment, the accuracy data and the reaction times should provide similar but independent measures of target discrimination. Thus, the mean reaction times should have a high degree of correlation to the calculated sensitivities for the human subjects. This allows the use of the mean reaction times as a continuous (non-censored) independent variable upon which to evaluate the effect of censored dependent sensitivities on ordinary least squares (OLS) estimates. The degree to which the censoring of the dependent variable affects the consistency of an OLS estimate can be seen in comparing the results of the estimated slope

and intercept coefficients produced by the non-linear censored regression to those resulting from ordinary least squares. Thus, OLS and non-linear censored regression coefficients were calculated using an Excel spreadsheet and the MATLAB function (listed in Appendix S) for each of the modeled output results regressed on the mean reaction times for the respective scene/sensor combinations. In addition, the results of each of the mathematical models were not regressed on the Human sensitivities due to both the response and independent variables having censored data.

V. RESULTS

The data obtained using the matched filter and template matching models for the various sensor by scene combinations are graphically summarized in Appendix N. Figures 13 and 15 presented examples of these summaries for the monochrome fused "cylindrical tank" for the global matched filter developed for this scene. An initial impression concerning how well the matched filter or the template matching filter discriminates the target from the background noise can be obtained by the amount of separation existing between the estimated N and SN distributions for each scene/sensor/filter combination. The a priori belief that the local matched filter would "outperform" the global matched filter in each scene was not constantly apparent. In some cases, the local filter seemed to result in a further overlap of the two empirical distributions. In fact, a larger separation was observed for only one of the three scenes (those scene containing the cylindrical tank). The threshold criteria (Nt's and SNt's) developed by filter evaluation at each of the three target positions are also listed in Appendix N for each respective scene/sensor/filter combination.

Comparisons of the four different sensors for each of the scene/filter combinations used in this experiment have been summarized on nine different empirical ROC plots (Appendix O). These ROC plots graphically display the performances for both the global and local matched filters, as well as for the template matching performances. These are particularly interesting from the viewpoint that this graphically represents the information contained in the estimated N and SN distributions from the perspective of paired Pd and Pfa without any specific threshold criteria. As discussed earlier, when the sensitivity for

the filters are calculated, the response bias is also calculated; these are listed in Appendix P. The resulting response bias values obtained for all of the filters were roughly the same as those calculated for the human subjects. If these filters adequately model the HVS, then the specific point on the empirical ROC plots would roughly equal that of a human observer (or the filter for that matter).

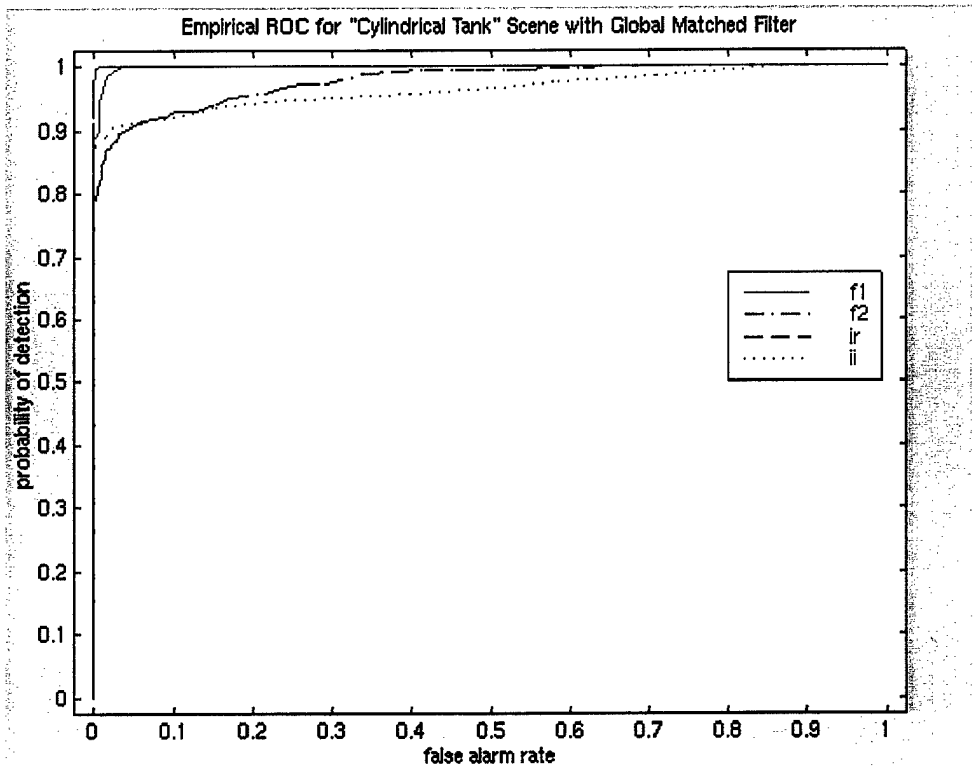


Figure 19. Empirically derived ROC plot illustrating the Global matched filter's sensitivities with the Cylindrical Tank scene for each sensor.

The results of the template matching do not model the HVS response biasing at all, but provides basic information about which sensor was the most difficult to find the target in. However, even this information generally corresponds to those bands that performed the poorest in the human tests.

The filter sensitivities resulting from these threshold criteria for the global (G), local (L), and template matching (T) filters are listed in Table 1. Also listed in Table 1 are the overall sensitivity results based on the accuracy data and the mean reaction times for the subject to detect the target (from the human factors experiment (Sampson, 1996)). Lastly, Table 1 also lists the corresponding output results of the contrast sensitivity function image quality metric (CSF IQM).

Sensor	Scene	Position	Global	Local	Template	Human	CSF IQM	Time (usecs)
F1	rct	1	3.139	0.325	4.653	4.020	0.372204	766.36
F2	rct	1	2.034	2.382	4.653	4.020	0.372303	726.96
IR	rct	1	4.653	3.571	4.653	4.020	0.372109	763.60
IL	rct	1	1.747	2.314	1.609	3.511	0.372471	761.84
F1	tnk	1	4.653	4.653	4.653	4.653	0.372220	617.49
F2	tnk	1	3.515	4.653	4.491	4.336	0.372385	659.00
IR	tnk	1	4.653	4.653	4.653	4.336	0.372300	583.38
IL	tnk	1	4.082	1.271	2.480	4.028	0.372249	664.47
F1	twr	1	3.477	2.107	4.653	4.336	0.372315	928.49
F2	twr	1	4.173	3.075	4.653	4.336	0.372430	715.58
IR	twr	1	1.945	3.123	4.008	2.466	0.372291	1201.44
IL	twr	1	4.132	3.222	4.653	4.336	0.372488	667.18
F1	rct	2	1.214	2.542	4.653	4.336	0.372261	723.33
F2	rct	2	1.722	2.673	4.653	3.437	0.372006	938.18
IR	rct	2	4.653	4.587	4.653	4.336	0.372361	747.40
IL	rct	2	1.112	1.814	1.756	3.547	0.372459	772.18
F1	tnk	2	4.653	4.653	4.653	4.653	0.372365	588.16
F2	tnk	2	4.551	4.653	4.491	4.653	0.372249	585.42
IR	tnk	2	4.653	4.653	4.653	4.653	0.372217	551.33
IL	tnk	2	3.497	1.202	2.674	4.653	0.372668	650.80
F1	twr	2	4.653	3.366	4.653	4.336	0.371843	599.13
F2	twr	2	4.241	3.680	4.653	4.336	0.372328	608.09
IR	twr	2	0.800	3.051	4.008	4.653	0.372159	686.73
IL	twr	2	4.132	3.740	4.653	4.653	0.372055	565.22
F1	rct	3	2.136	2.824	4.653	4.653	0.372364	781.67
F2	rct	3	2.280	2.603	4.653	4.020	0.372118	845.40
IR	rct	3	4.653	4.653	4.653	4.653	0.372228	698.49
IL	rct	3	1.342	1.941	1.655	4.028	0.372573	751.89
F1	tnk	3	4.653	4.653	4.653	4.653	0.372458	627.13
F2	tnk	3	3.257	4.653	4.491	4.653	0.372373	585.82
IR	tnk	3	4.653	4.653	4.653	4.336	0.372184	615.96
IL	tnk	3	1.984	0.994	3.025	4.336	0.372377	757.11
F1	twr	3	4.152	3.583	4.653	4.653	0.372253	673.67
F2	twr	3	4.447	3.121	4.653	4.653	0.372463	619.80
IR	twr	3	1.719	2.987	4.008	4.336	0.372249	803.56
IL	twr	3	3.977	3.867	4.653	4.653	0.371877	594.42

Table 1. This table summarizes the sensitivity results for the global and local matched filters, the sensitivities for the template matching filter, and the overall results of the subjects participating in the human factors experiment (Sampson, 1996). Also listed are the corresponding output values from the contrast based IQM and the mean reaction times from the human factors experiment. Shaded cells contain sensitivity values which have been censored to a maximum value of 4.6527 (rounded to 4.653)

In order to better illustrate the relationship between these sensitivities and its range of detection and false alarm rates, the four sensitivities of each row may be plotted on a single ROC plot.

An example of these results is shown in Figure 20 and listed in Appendix Q in the same order as in Table 1. The sensitivities from the filters represent the quantitatively modeled sensitivity performances of each filter. The HVS sensitivities represent the actual qualitative sensitivity performances of the human observers.

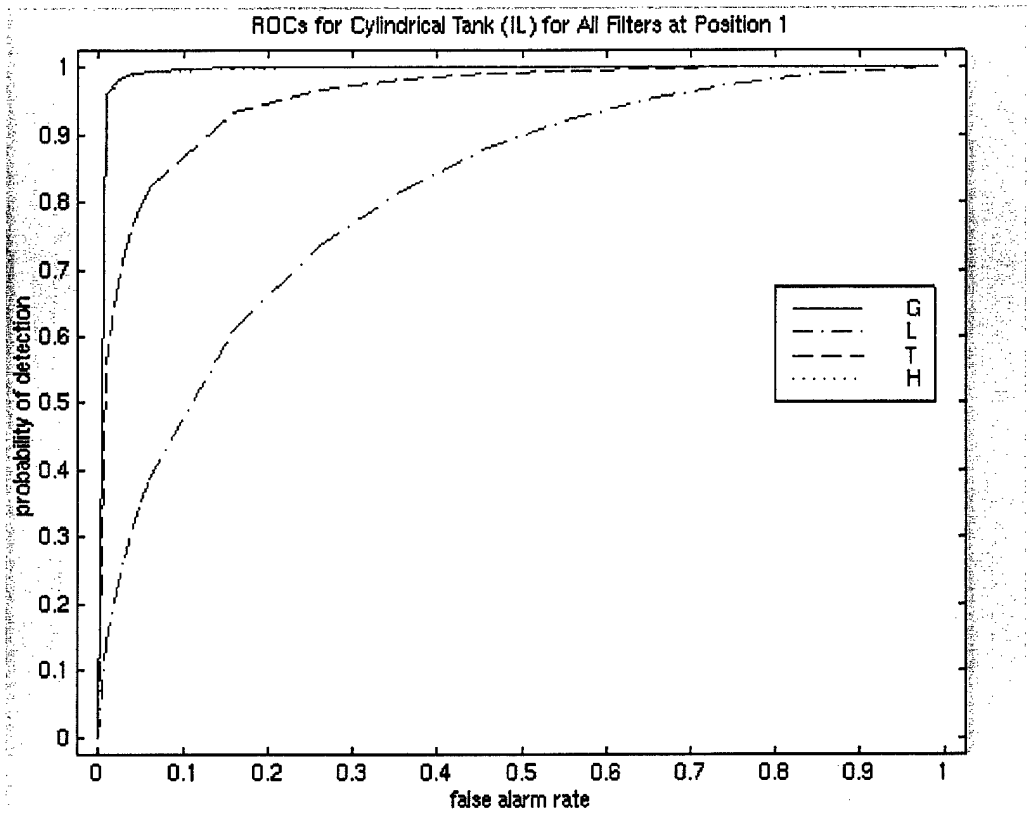


Figure 20. ROC plot based on calculated sensitivities from the Global and Local matched filters, the Template matching filter, and the overall sensitivity results of the human subjects.

The sensitivity results were modeled with the sensor, scene, position, and filter as main effects, plus specific two-way and three-way interactions. The three two-way interactions included in this model were sensor by scene, scene by filter, and position by

filter. The two three-way interactions included in this model were sensor by scene by position, and sensor by scene by filter. The summary statistics for the main effects, the two-way, and three-way interactions are listed in Tables 2 through Table 10. These numbers reflect the mean (and variance) of the sensitivities corresponding to the main effect or interaction shown.

Filter			
G	H	L	T
3.370311 (1.718986)	4.385982 (0.333780)	3.235733 (1.501949)	4.17665 (0.88853)

Table 2. Mean (variance) of sensitivities by filter.

Sensor			
F1	F2	IL	IR
4.265946 (0.749956)	4.167288 (0.603163)	3.725834 (1.369461)	4.197236 (0.819675)

Table 3. Mean (variance) of sensitivities by sensor.

Scene		
rct	tnk	twr
3.781346 (1.310651)	4.335793 (0.608770)	4.150089 (0.704194)

Table 4. Mean (variance) of sensitivities by scene.

Position		
1	2	3
3.954007 (1.021190)	4.129856 (0.934112)	4.183365 (0.800179)

Table 5. Mean (variance) of sensitivities by position.

	Scene		
Filter	rct	tnk	twr
G	2.556975 (1.887544)	4.066733 (0.720071)	3.487225 (1.596225)
H	4.220105 (0.409969)	4.542125 (0.111898)	4.395717 (0.437916)
L	2.685458 (1.402824)	3.778417 (2.505373)	3.243325 (0.219060)
T	3.907808 (1.816951)	4.130567 (0.735885)	4.491575 (0.084964)

Table 6. Mean (variance) of sensitivities for filter by scene.

	Position		
Filter	1	2	3
G	3.516683 (1.176958)	3.323292 (2.579346)	3.270958 (1.676644)
H	4.203833 (0.569456)	4.430417 (0.260034)	4.523697 (0.128130)
L	2.945475 (1.850193)	3.384300 (1.367333)	3.377425 (1.423513)
T	4.150733 (1.036966)	4.179167 (0.914447)	4.200050 (0.874407)

Table 7. Mean (variance) of sensitivities for filter by position.

	Scene		
Sensor	rct	tnk	twr
F1	3.858747 (1.544557)	4.652698 (3.9e-12)	4.286393 (0.440988)
F2	3.680562 (1.038419)	4.476560 (1.657e-1)	4.344743 (0.277942)
IL	3.119437 (1.585758)	3.653359 (1.587171)	4.404706 (0.184445)
IR	4.466638 (0.280758)	4.560555 (9.745e-2)	3.564513 (1.520873)

Table 8. Mean (variance) of sensitivities for sensor by scene.

	Scene by Position 1		
Sensor	rct	tnk	twr
F1	3.64600 (2.19455)	4.652697 (2.285e-12)	4.049222 (0.882101)
F2	3.765083 (1.165162)	4.351959 (2.601e-1)	4.257272 (0.3845237)
IL	3.144903 (1.411401))	3.610608 (1.462868)	4.270559 (0.3388849)
IR	4.241008 (0.6721999)	4.514484 (1.528e-1)	2.944284 (1.3150168)

Table 9.1. Mean (variance) of sensitivities for sensor by scene by position 1.

	Scene by Position 2		
Sensor	rct	tnk	twr
F1	3.820746 (1.7097588)	4.652697 (4.285e-12)	4.353584 (0.3091028)
F2	3.453208 (1.1881292)	4.619772 (3.978e-3)	4.341396 (0.2233477)
IL	2.964662 (1.7549734)	3.829447 (1.677025)	4.473535 (0.1210126)
IR	4.506209 (0.1507578)	4.652697 (4.285e-12)	3.890310 (1.8819933)

Table 9.2. Mean (variance) of sensitivities for sensor by scene by position 2.

	Scene by Position 3		
Sensor	rct	tnk	twr
F1	4.109485 (1.045441)	4.652697 (4.285e-12)	4.456372 (0.1552901)
F2	3.823395 (9.681e-1)	4.457947 (2.388e-1)	4.435560 (0.2871857)
IL	3.248745 (1.996747)	3.520021 (2.017254)	4.470022 (0.1152815)
IR	4.652697 (4.286e-12)	4.514484 (1.528e-1)	3.858946 (1.140124)

Table 9.3. Mean (variance) of sensitivities for sensor by scene by position 3.

	Sensor by Scene="Rectangle"			
Filter	F1	F2	IL	IR
G	2.162800 (0.9269369)	2.012033 (0.07824410)	1.400367 (0.103098663)	4.652700 (0.0000000)
H	4.431554 (0.2095869)	4.045513 (0.47688795)	3.971799 (0.462484366)	4.431554 (0.3842423)
L	1.896700 (1.8732126)	2.552200 (0.02308237)	2.023000 (0.067380430)	4.269933 (0.3678959)
T	4.652700 (0.0000000)	4.652700 (0.00000000)	1.673133 (0.005688403)	4.652700 (0.0000000)

Table 10.1. Mean (variance) of sensitivities for filter by sensor by scene ("Rectangle").

	Sensor by Scene="Cylindrical Tank"			
Filter	F1	F2	IL	IR
G	4.652700 (0)	3.774167 (0.46956737)	3.187367 (1.17243466	4.652700 (0.0000000))
H	4.652696 (0)	4.578982 (0.08150601)	4.431554 (0.20958689)	4.505268 (0.1513683)
L	4.652700 (0)	4.652700 (0.00000000)	1.155567 (0.02086860)	4.652700 (0.0000000)
T	4.652700 (0)	4.490700 (0.00000000)	2.726167 (0.07644562)	4.652700 (0.0000000)

Table 10.2. Mean (variance) of sensitivities for filter by sensor by scene ("Cylindrical Tank").

	Sensor by Scene="Tower"			
Filter	F1	F2	IL	IR
G	4.093800 (0.3482844)	4.286933 (0.0203720)	4.080333 (0.007977363)	1.487833 (0.367387403)
H	4.505268 (0.1513683)	4.505268 (0.1513683)	4.578982 (0.081506012)	3.993348 (1.226095276)
L	3.018300 (0.6352025)	3.291967 (0.1133516)	3.609700 (0.116682210)	3.053333 (0.004602443)
T	4.652700 (0.0000000)	4.652700 (0.0000000)	4.652700 (0.0000000)	4.008200 (0.0000000)

Table 10.3. Mean (variance) of sensitivities for filter by sensor by scene ("Tower").

A. ANOVA/PLANNED COMPARISONS ANALYSIS

All four of the main effects were significant. The significance of sensor ($F(3,210)=17.42$, $p<0.00001$), showed the fused color sensor to have the highest sensitivity, followed by the infrared, the fused monochrome, and the intensified light sensors. The fused algorithm sensitivities were similar to the infrared band. A priori contrast showed that the fused monochrome sensor sensitivity was significantly different from those of the fused color, infrared, and intensified light sensors, $F(1,210)=46.36$, $p<0.00001$. This may have been a reflection of the notably different sensitivity results between the intensified light and the fused monochrome. The significance of scene ($F(2,210)=30.66$, $p<0.00001$), showed that some of the scenes, particularly the Rectangle scene, were not as accommodating to detecting the target as were the Tower or Cylindrical Tank scenes. This was evident in the mean reaction times collected in the human factors experiments (Sampson, 1996). The significance of filter ($F(3,210)=81.48$, $p<0.00001$), showed that the most discriminating and accurate "filter" was the HVS with the highest sensitivity. This was followed by the Template matching filter, then the Global and Local matched filters respectively. It is not particularly surprising that the Template matching filter produced a higher degree of accuracy than the matched filters. As explained previously, the Template matching filter looks for an exact digital match. Since this experiment provided the exact target to be located, the Template matching filter picked out the target perfectly without variance. A priori contrasts across filters showed that the Global matched filter differed significantly from the Local matched filter, $F(1,210)=38.75$, $p<0.00001$). While the Global and Local matched filter differed

significantly, $F(1,210)=175.2$, $p<0.00001$, indicating that sensitivity is influenced by the specified range of the scenes. Furthermore, both the Global and Local matched filters differed significantly from the Template matching filter, $F(1,210)=30.49$, $p<0.00001$). The significance of position ($F(2,210)=5.54$, $p<0.005$) reflects the different target threshold criteria developed at each of the three target positions in each of the three scenes. However, as the results of the two- and three-way interactions are presented, target position becomes non-significant.

The two-way interaction between sensor and scene was significant, ($F(6,210)=23.76$, $p<0.00001$). This shows that regardless of the main effects of sensor and scene, the sensor producing the highest sensitivity was highly dependent upon scene. This is not surprising since the background content of each scene differs characteristically from each of the others', resulting in a wide image quality range for any sensor from one scene to the next. A priori contrasts applied across specified sensors again showed a significant difference between the sensitivity results of the fused monochrome sensor and the other sensors, $F(2,210)=17.28$, $p<0.00001$. This significant difference from either the infrared or the intensified light sensors was due to one of these single band sensors being significantly poorer than the other.

Another significant interaction existed between scene and filter, $F(6,210)=81.48$, $p<0.00001$. This result showed that the best filter was highly dependent upon the scene used. A priori contrasts applied across specified filters with the various scenes showed that, regardless of the interaction between scene and filter, both the Global and Local matched filter sensitivities still differed significantly from the sensitivities obtained through

human factors testing, $F(2,210)=14.36$, $p<0.00001$. However, in the sensor by filter interaction, the Global matched filter sensitivities were not significantly different from the Local matched filter sensitivities, $F(2,210)=0.1072$, $p<0.8984$. As before, both the Global and Local matched filter sensitivities remain significantly different from the Template matching sensitivities, $F(2,210)=7.59$, $p<0.0007$. The last two-way interaction included in this model, position by filter, was not significant, $F(6,210)=1.61$, $p<0.1445$. The insignificance of this interaction was due to each filter's very similar sensitivities resulting from each of the three target positions used.

The three-way interaction for sensor by scene by position essentially compared the sensitivity results from one target position with those of another position within the same sensor by scene combination. There was not enough evidence to show that the sensitivities evaluated at the three different target positions within the same scene were significantly different from one another, $F(22,210)=1.14$, $p<0.3073$. A priori contrasts comparing the fused color sensor sensitivity results to those of the infrared and intensified light sensors provided significant evidence that these sensors had the same sensitivity results, $F(4,210)=0.09058$, $p<0.98534$. However, the last interaction for sensor by scene by filter indicated that each of the filters performed significantly different for each of the 12 scene/sensor images, $F(27,210)=9.64$, $p<0.00001$. Additionally, the a priori contrasts developed across specified filters further showed that the sensitivity results of the Global and Local matched filters differed significantly from those derived from the human factors experiment for the same sensor by scene images, $F(6,210)=12.06$, $p<0.00001$. In addition, the Global matched filter sensitivity results differed significantly from those produced by

the Local matched filter for the same scene/sensor images, $F(6,210)=14.82$, $p<0.00001$. Also, the Global and Local matched filter sensitivities were significantly different from those of the Template matching filter, $F(6,210)=9.93$, $p<0.00001$. A priori contrasts developed across specified sensors further showed that the sensitivities of the fused monochrome sensor differed significantly from those of the other sensors, $F(6,210)=22.16$, $p<0.00001$.

B. BONFERRONI TESTS FOR COMPARISONS

Investigation of significant results found from the planned comparisons above further specified contrasts with sensitivity as the response variable. Listed in Tables 11 through 14 are the resulting 95 percent simultaneous confidence intervals for the specified linear combinations, by the Bonferroni method with Sensitivity as the response variable (intervals excluding 0 are flagged by '****'). The results of this post hoc analysis further compare the elements of the main effects of the ANOVA/Planned comparisons analysis in a pair-wise fashion.

	Estimate	*Std.Error*	*Lower Bound*	*Upper Bound*
G-H ****	-1.020	0.0912	-1.2600	-0.773
G-T ****	-0.806	0.1180	-1.1200	-0.493
H-L ****	1.150	0.0912	0.9070	1.390
H-T	0.209	0.0912	-0.0335	0.452
L-T ****	-0.941	0.1180	-1.2500	-0.627

Table 11. Results of the Bonferroni comparisons between Filter sensitivities (critical point: 2.6635)

	Estimate	*Std.Error*	*Lower Bound*	*Upper Bound*
1-2	-0.1250	0.0912	-0.345	0.0949
1-3	-0.1390	0.0912	-0.359	0.0812
2-3	-0.0137	0.0912	-0.234	0.2060

Table 12. Results of the Bonferroni comparisons between Position sensitivities (critical point: 2.4133)

The results of the Bonferroni comparisons for Filter (see Table 11) shows the resulting comparisons between the sensitivities for each of the filters. The Template matching filter provided sensitivity values closest, in mean, to the Human sensitivities. However, due to the extreme digital precision with which this filter operates, it should not be a good approximation of the HVS responses. This should become apparent in regressing the Template matching sensitivity results on the qualitative Human data. Table 12 shows the results of Bonferroni comparisons for the elements of the Position main effect. The results imply that the sensitivity results of the three different target positions presented did not necessarily impact the sensitivities for the various sensor by scene by filter combinations. This may not have been apparent in the ANOVA main effect for Position due to the large number of interactions incorporated into the model. Table 13 further shows that the significance of the Scene main effect resulting from the ANOVA and Planned Comparisons were not due to a singular difference in only one of the three scenes, but, rather, due to all the scenes being different. The results displayed in Table 14 shows that that the significance existing for the Sensor main effect was, in fact, primarily due to the sensitivities resulting from the single-band IL sensor and is apparent in Table 3 or the summary statistics presented earlier.

	Estimate	*Std.Error*	*Lower Bound*	*Upper Bound*
rct-tnk ****	-0.787	0.0912	-1.01000	-0.567
rct-twr ****	-0.562	0.0912	-0.78200	-0.342
tnk-twr ****	0.225	0.0912	0.00497	0.445

Table 13. Results of the Bonferroni comparisons between Scene sensitivities (critical point: 2.4133)

	Estimate	*Std.Error*	*Lower Bound*	*Upper Bound*
F1-F2	0.0441	0.105	-0.236	0.324
F1-IL ****	0.8780	0.105	0.597	1.160
F1-IR	-0.0824	0.105	-0.363	0.198
F2-IL ****	0.8340	0.105	0.553	1.110
F2-IR	-0.1260	0.105	-0.407	0.154
IL-IR ****	-0.9600	0.105	-1.240	-0.680

Table 14. Results of the Bonferroni comparisons between Sensor sensitivities (critical point: 2.6635)

C. TREND ANALYSIS WITH CENSORED DATA

The results of using the non-linear censored regression to approximate the MLE's for the log-likelihood for each of the modeled results are listed in Appendix S and summarized in Table 15 below. The significance of the MLE coefficients was obtained by calculating the t-statistic for each of the coefficients. The t-statistic is calculated by dividing the MLE coefficients by their associated standard deviations. The only regressions that appear to be non-significant are the Template matching model and the contrast-based IQM model.

	H~Time	H~CSFIQM	G~Time	L~Time	T~Time
Intercept (Variance)	5.982633 (0.004189)	9.097626 (1499.75)	6.210681 (0.0502)	3.56043 (.048387)	3.53893 (2.39919)
Slope (Variance)	-0.00256 (7.107e-9)	-12.92669 (10821.01)	-0.00455 (9.06e-8)	-0.00117 (8.508e-8)	-4.248e-4 (3.459e-6)

Table 15. Summary of the non-linear MLE estimates of the regression coefficients for the regression models (columns names). The letters are abbreviations for Human (H), Global (G) matched filter model, Local (L) matched filter model, Template (T) matching model, and CSFIQM for the contrast based IQM model.

The results of the OLS regressions for the same models listed in Table 15, are summarized in Table 16.

	H~Time	H~CSFIQM	G~Time	L~Time	T~Time
Intercept (p-value)	6.30369 (1.48e-22)	9.097626 (0.95745)	7.43154 (2.627e-8)	6.13259 (1.568e-6)	5.03044 (2.386e-6)
Slope (p-value)	-0.00286 (7.444e-9)	-12.92669 (.977489)	-0.00575 (3.352e-4)	-0.004101 (0.00872)	-0.001209 (0.336)
Regression Significance F	7.44e-09	0.855	3.35e-04	8.72e-03	0.336
R-squared	0.631	0.000023	0.319	0.186	0.027

Table 16. Summary of the coefficients estimated using OLS regression and the respective p-values for the regression models (column names). The letters are abbreviations for Human (H), Global (G) matched filter model, Local (L) matched filter model, Template (T) matching model, and CSFIQM for the contrast based IQM model.

For those regressions with significant coefficients, the MLE coefficients and the OLS coefficients will be nearly the same if the effect of censoring the dependent variable is minor. Thus, the effects of censoring are minor in the Human~Time as well as the Gobal~Time regression models.

As predicted, the regression results for the Template matching model were non-significant. Although the Template matching sensitivities have a similar mean to the

Human sensitivities, the coefficient of correlation of only two percent suggests that this model fails to capture the "trends" represented in the Human data. The regression results of the contrast-based IQM was also non-significant. There appears to be sufficient evidence to suggest that the contrast-based IQM had no correlation to the trends represented in the Human data. In the case of the Local~Time regression, the censoring of the dependent variable had a major effect. Thus, the OLS regression results for the Local~Time regression were inconsistent and could not be used to infer any relationship between the Local matched filter sensitivity data and the Human data.

VI. CONCLUSIONS

The area of sensor fused imagery has a large and diverse number of fusion algorithms and approaches to "improve" night vision displays. This thesis addressed the problem of quantifying the degree of enhancement achieved by fusion algorithms, based upon mathematical models of the HVS and search and detection theory. The four a priori hypotheses presented in the thesis background were developed to evaluate whether a local and global matched filter model, as well as a contrast based IQM model, could replicate the results of human subjects for the same "natural" and "coherent" scenes using actual targets.

Based upon the results of regression analysis, there is significant evidence that there are large differences between the sensitivities produced by the Template matched filter and the results of the human data. The contrast-based IQM also demonstrated large differences between its estimated sensitivity performances of the various scenes and the results of the human data. However, in the case of the Global and Local matched filters, the resulting sensitivities tend to capture some of the trends seen in the data collected from the human factors experiment (Sampson, 1996). Of the four models evaluated, the Global matched filter produced the highest degree of correlation with the human data. These results are particularly important in that Sampson carried out the human factors experiment with the expectation that the fused colored images would produce the shortest reaction times in each of the scenes presented. He based this expectation on the superior results predicted by the Global matched filter for a simulated target within a completely different background image. As the results of the large number of significant interactions

across scene indicate, it is not possible to make this generalization. However, as can be seen by the empirically derived ROC plots for the Cylindrical Tank and Tower scenes with the Global matched filter (see Appendix R) with the targets placed in their original positions (position 1), the estimated performances of the four different sensors track well with the mean reaction times collected from the human factors experiment. Most importantly, these empirical ROC plots of the Global matched filter correctly identified that the fused color or fused monochrome image would not outperform the single band infrared or intensified light band for that target and background imagery. Although there are significant differences based upon the ANOVA, this is more of a reflection of the differences in the sensitivity "operating range" of the four different models which is a reflection of the obvious differences apparent in the summary statistics presented earlier.

Despite the similarities of the Global and Local matched filter development, there are still significant differences in the predicted sensitivity results for the same images. This was most apparent from the ANOVA, which showed that these two matched filters are significantly different. However, the a priori expectation that the Local matched filter should outperform the Global matched filter does not appear to be the case. Based upon the higher degree of correlation between the Global matched filter and the Human data, truncating the spatial image, and consequently the spectral content, of the background used to develop the matched filter coefficients does not increase performance.

The inclusion of the Template matching filter provided a basis for comparing the Global and Local matched filters (which model the HVS) with a standard image processing search filter (which does not model the HVS). The resulting statistical analysis

showed that there was almost no relationship between the sensitivities predicted by the Global or Local matched filters and the Template matching filter for the respective scenes. Since the Global and Local matched filters follow the sensitivity trends of the Human data rather than the Template matching filter, the Global or Local matched filter process appears to be a viable model of the HVS.

Although the degree to which the Global or Local matched filter sensitivities correlate to the Human data is not accurate in all cases, the Global matched filter moderately predicted which of the single-band sensors (infrared or intensified light), as well as which of the fused sensors (color-fused or monochrome-fused) provided the higher sensitivities in the majority of the cases. Thus, it still might be possible to use the Global matched filter concept developed for this experiment to evaluate and compare the many different fusion algorithms being proposed.

A number of factors could be handled differently and the effects these variations have on the accuracy of the sensitivities predicted by the Global matched filter investigated. The empirical distributions shown in Appendix N reflect that, in the majority of the scenes, the noise and signal+noise distributions have unequal variances. The search and detection procedure used for this experiment assumed equal variances when calculating the filter sensitivities. However, the scenes that provided the highest degree of correlation with the human data are those that tended to display fairly equal variance in the empirical histograms. A procedure exists which allows the calculated sensitivities to be "scaled" to account for unequal variances is covered extensively in Gescheider (1985).

Also, in the development of the matched filters, certain assumptions may have affected the resulting matched filter coefficients to some degree. When the matched filters were created, the filter was enlarged proportionally, based upon the size of the target. The amount of background to include in the filter evaluations could arguably be of a higher proportion. However, the enlarged filter was kept rather limited because, as filter size increases, the processing time required to carry out even a suboptimal sampling of the 36 scenes processed increases exponentially. Additionally, the threshold criteria used in developing the sensitivities from the empirical distributions were chosen to reflect how well the matched filters could distinguish the target from the background at the same locations used in the human factors experiment. The empirical ROC plots listed in Appendix O still reflect the same "trends" as the Human data without a specified threshold criterion. Since the suboptimal sampling of the scene allows the estimation of the noise and signal+noise distributions, the sensitivity of a particular scene could be calculated directly from the difference in the means of these distributions while scaling for unequal variances.

The limited number of scenes available for these evaluations may have also contributed another component of uncertainty. This is because all the scenes used for this and the human factors experiment were digitally "processed" to create the same scenes with and without targets at the three different locations. This was accomplished by taking scenes in which the target existed already and digitally moving the target to other locations within the scene. In order to produce this same scene without a target present, a section of background "visibly" similar to the target area was cropped and placed over the target.

While this may be appropriate for the human factors experiment, it is definitely not arguably sound for evaluations with digital filters. Datasets of images need to be collected, some with targets already existing in the various locations as well as without the target present. This would also remove any issues concerning the influences of luminance and reflectivity of the target's appearance in different locations.

Of the four models evaluated in this experiment, the Global matched filter came out as the best candidate for predicting the target detection abilities of the human visual system. However, there are many refinements which need to be explored in order to increase the degree of correlation with the human factors data. The fact that the contrast-base IQM performed so poorly attests to the difficulties in incorporating the human visual system into standard image processing techniques. However, the matched filter process appears to be a viable model of the human visual system.

APPENDIX A. MATCHED FILTER MAIN PROGRAM (MATLAB)

Filename: **go_MCfilter.m**

```
%
% MAIN PROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%
%   This is the Main Program of the Matched Filter program.  It "coordinates"
% the main program steps.  The following is a list of Variables used throughout
% the m-file code.
%
%
%   b     : holds the original background image loaded for the blue component.
%   b_mean : holds the mean of the specified field of interest for the blue
%            component image.
%   b_std  : holds the standard deviation of the specified field of interest
%            for the blue component image.
%   bkgd_r : assigned the red background (or the only background to be
%            filtered) once any preprocessing has been carried out on the
%            original background image (variable r).
%   bkgd_g : assigned the green background once any preprocessing has been
%            carried out on the original background image (variable g).
%   bkgd_b : assigned the blue background one any preprocessing has been
%            carried out on the original background image (variable b).
%   BNtype : either [N]ormalize the background image or leave [U]nnormalized.
%   Choice : [F]ield filtering option or [P]oint filtering option
%   Filtertype: [L]ocal or [G]lobal matched filters or [T]emplate matching.
%   fil_NNR_2db: "Noise" output value for h_b filter at the specified point.
%   fil_NNR_2dg: "Noise" output value for h_g filter at the specified point.
%   fil_NNR_2dr: "Noise" output value for h_r filter at the specified point.
%   fil_NNR_3d: Summed "Noise" output value from h1, h2, and h3 3D filters.
%   fil_SNR_2db: "Signal+Noise" output value for h_b filter at the specified
%              point.
%   fil_SNR_2dg: "Signal+Noise" output value for h_g filter at the specified
%              point.
%   fil_SNR_2dr: "Signal+Noise" output value for h_r filter at the specified
%              point.
%   fil_SNR_3d: Summed "Signal+Noise" output value for h1, h2, and h3 3D
%              filters.
%   Foptions: string made up of the filter options: [Process, BNtype, TNtype,
%              Filtertype, Choice, (F or P)Series] ie: "1NNGF1"
%   FSeries : Field "series" are numaerically ordered integers for multiple
%             fields to be saved with the same filename conventions.
%   g     : holds the original background image loaded for the green
%            component image.
%   g_mean : holds the mean of the specified field of interest for the green
%            component image.
%   g_std  : holds the standard deviation of the specified field of interest
%            for the green component image.
%   h1    : the 3D filter coefficients for the red component.
```

```
%   h2    : the 3D filter coefficients for the green component.
%   h3    : the 3D filter coefficients for the blue component.
%   h_r   : the 2D filter coefficients for the red component.
%   h_g   : the 2D filter coefficients for the green component.
%   h_b   : the 2D filter coefficients for the blue component.
%   h_123_dim: [row,col] dimension of the 3D red, green, and blue filters.
%   h_rgb_dim: [row,col] dimension of the 2D red, green, and blue filters.
%   Imagename: holds a name string of the background image.
%   index: an sequentially numbered vector of numbers from 1 to the number of
%          points within the specified field which have been evaluated.
%   keep_b : [Y]es to indicated keeping the blue component or [N]o.
%   keep_C : [Y]es to indicated keeping the Color component or [N]o.
%   keep_g : [Y]es to indicated keeping the green component or [N]o.
%   keep_r : [Y]es to indicated keeping the red component or [N]o.
%   m     : holds the number of rows in the background image.
%   m_offset: holds the row offset when the background image is truncated to a
%          smaller area withing the original image (such as in the
%          case of a locally derived matched filter).
%   mm    : holds the half-height of the background image.
%   n     : holds the number of columns in the background image.
%   n_offset: holds the column offset when the background image is truncated
%          to a smaller area within the original image (such as in the
%          case of a locally derived matched filter).
%   noise_1: assigned the "noise" only values of bkgd_r filtered with h1.
%   noise_2: assigned the "noise" only values of bkgd_r filtered with h2.
%   noise_3: assigned the "noise" only values of bkgd_r filtered with h3.
%   noise_r: assigned the "noise" only values of bkgd_r filtered with h_r.
%   noise_g: assigned the "noise" only values of bkgd_r filtered with h_g.
%   noise_b: assigned the "noise" only values of bkgd_r filtered with h_b.
%   nn    : holds the half-width of the background image.
%   o***  : when placed before a variable in this list, it is used to hold
%          the original value of the variable since it will be changing.
%   output1: assigned the "Signal+Noise" contributions from the red component.
%   output2: assigned the "Signal+Noise" contributions from the green
%          component.
%   output3: assigned the "Signal+Noise" contributions from the blue component.
%   Process: holds the chosen filter processing options.
%          [1] Process a single "color" image (includes fused monochrome).
%          [2] Process multiple bands iot predict fused image performance.
%          [3] Process a color fused image by its RGB components.
%   PSeries : Point "series" are numaerically ordered integers for multiple
%          points to be saved with the same filename conventions.
%   r     : holds the original background image loaded for the red component.
%   r_mean : holds the mean of the specified field of interest for the red
%          component image.
%   r_std  : holds the standard deviation of the specified field of interest
%          for the red component image.
%   scene_r: assigned a filter-sized image segment of the red background image
%          for use in convolving with the h_r or h1 filters.
%   scene_g: assigned a filter-sized image segment of the red background image
%          for use in convolving with the h_g or h2 filters.
%   scene_b: assigned a filter-sized image segment of the red background image
%          for use in convolving with the h_b or h3 filters.
%   summary: a two dimensional dataframe storing the "Noise" and
```

```
%           "Signal+Noise" output values of each of the filters at each
%           point (specified by the chosen filtering options) in the format:
%      [index, fil_SNR_2dr, fil_SNR_2dg, fil_SNR_2dg, fil_SNR_3d,
%       fil_NNR_2dr, fil_NNR_2dg, fil_NNR_2dg, fil_NNR_3d, row#, col#]
%   sum_dim: the [row,col] dimensions of the 2-D dataframe holding the
%   t_r   : holds the target image for the red (or only band specified) image.
%   t_g   : holds the target image for the green image.
%   t_b   : holds the target image for the blue image.
%   Targetname: holds a name string of the target image.
%   Tm1   : holds the number if rows in the target image.
%   Tm2   : holds the half-height of the target image.
%   Tm3   : holds the half-height of the filter.
%   Tm4   : holds the number of rows in the filter.
%   Tn1   : holds the number of columns in the target image.
%   Tn2   : holds the half-width of the target image.
%   Tn3   : holds the half-width of the filter.
%   Tm4   : holds the number of columns in the filter.
%   TNtype : holds either [N]ormalize the target image or leave [U]nnormalized.
%   Tr_template: holds the target template for used with t_r.
%   Tg_template: holds the target template for used with t_g.
%   Tb_template: holds the target template for used with t_b.
%   UniformCheck: an attempt to determine if the target to be used is a
%           uniform grey-level such was the case with some of the
%           synthetic targets used for run checks.
%   Xmax  : maximum column number of the field specified to be filtered.
%   Xmin  : minimum column number of the field specified to be filtered.
%   Ymax  : maximum row number of the field specified to be filtered.
%   Ymin  : minimum row number of the field specified to be filtered.
%
%
%
%
%
%
%
%    These include calls to other subroutines summarized as follows:
%
%
% "get_images": This m-file allows a choice of Processing options which then
%    determines the number and types of TIFF images which need to be openned.
%    Once this has been specified this routine uploads the three image files
%    required, (1) background image, (2) target image, and (3) a target
%    template image.  More than one image set of three may be used in the
%    case of processing options (2) and (3).
%
% "get_filt_images": This m-file gets the inputs for the various filtering
%    options:
%      (1) Preprocess (normalize) the background or target, both, or
%    none.
%      (2) Choice of filter type: (G)lobal matched filter, (L)ocal matched
%    filter, or (T)emplate matching
%      (3) Area within image to be processed: (F)ield or (P)oint
%    Note:  In order to use the (P)oint option, the (F)ield option would have
%    already been run since the (P)oint option uses the filter coefficients
```

```
%     derived and saved as well as the same preprocessing options.  Thus, when
%     (P)oint is specified, the program looks for the file it created and
%     stored the filter coefficients for the currently specified background
%     image, target, preprocessing, and Field serial number.
%
% "do_preprocessing":  This m-file carries out any specified normalization to
%     the background(s) and target(s) as well as truncates the background to
%     be used in developing a locally matched filter and "local" evaluations.
%
% "filter_image":  This m-file carries out filtering the background image(s)
%     with the developed filter(s) at the specified point or at the specified
%     horizontal and vertical step sizes within the specified field in a
%     sequential manner.
%
% "conv_image":  This m-file does the same filtering process as filter_image
%     but does this by inserting targets at several locations at once.
%     However, the speed advantage of going this route depends on the size of
%     the target with respect to the specified field to be filtered.  In
%     certain situations where the number of targets which can be fit within
%     the specified field are above 300 or more, this method should save time.
%
% "roc9":  This m-file creates an empirical ROC plot of the individual bands
%     as well as any specified "fused" band results and saves the plot summary
%     for recall using the recall_roc9 m-file.
clear
get_images;
get_filt_info;
do_preprocessing;

if (Filtertype=='L' | Filtertype=='G') & (Choice=='F')
 make_GLfilters
elseif (Filtertype=='T') & (Choice=='F')
 make_Tfilters
end;

if ((Filtertype=='L')|(Filtertype=='T'))&(Choice=='F')
 if ((Xmax-Xmin)/Tn4)*((Ymax-Ymin)/Tm4)>200
  conv_image
 else
  filter_image
 end;
else
 filter_image
end;

if Choice=='F'
 roc9;
end;
```

APPENDIX B. M-FILE TO LOAD IMAGES AND TARGETS

filename: **get_images.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%
%   This MATLAB m-file is a sub-program of "go_MCfilter".  This routine
% allows specification of the type if "processing" which is to be carried
% out on and image(s).  It is written to handle TIFF image files saved with
% either the "indexed" or "RGB" option.
%     Processing options:
%         [1] If the first process is chosen, then the image and target are
% converted to a grayscale image and filtered.
%         [2] If the second process is chosen, then a different sensor/image
% may be entered for each of the color bands (up to three: red, green, and
% blue).  For instance, a long-wave IR image/target can be designated to the
% red component, a medium-wave IR image can be designated to the green
% component, and a short-wave IR image can be designated to the blue component.
% This processing option will calculate the filters for each of the bands and
% filter the image as specified for the individual band results.  It will then
% calculate the 3-D results of fusing the component images using 3-D filters.
%         [3] If the third option is chosen, then a pseudo-color image
% should be specified and the image will be split into its RGB components and
% if redundant components exit, then an option to specify which components to
% keep will appear since the results of redundant bands will be identical.
% This option then processes the individual components and then the fused
% 3-D result as in option two.
%
%
% Uses: (other m-files)
%      choose_components.m
%      get_file.m
%
%

disp(' ');
disp(' ');
disp(' ');
disp(' ');
disp(' Matched filter processing options:');
disp(' ');
disp('    [1] Process a single "color" image (includes fused monochrome).');
disp(' ');
disp('    [2] Process multiple bands iot predict fused image performance.');
disp(' ');
disp('    [3] Process a pseudo-color fused image by its RGB components.');
disp(' ');
disp(' ');
```

```
Process=input('Enter Processing option: ');
disp(' ');
if Process==2
 disp(' ');
 disp(' ');
 Nonsns=1;
 msg='N'; keep_r='*'; keep_g='*'; keep_b='*'; keep_C='Y';
 [keep_r,keep_g,keep_b]=choose_components(Nonsns,Nonsns,Nonsns,msg);
 clear msg Nonsns
 TProcess=1; dumb1=0; dumb2=0; dumb3=[]; FusedImage=[]; Fusedtarget=[]; m=0; n=0;
 r=[]; g=[]; b=[]; t_r=[]; t_g=[]; t_b=[]; Tr_template=[]; Tg_template=[]; Tb_template=[];

 if keep_r=='Y'
  disp(' ');
  disp(' ');
  disp('Enter the name of the RED BACKGROUND image to be used (".tiff") assumed');
  Imagename=input(':: ','s');
  [FusedImage,m,n,r,dumb1,dumb2]=get_file(FusedImage,Imagename,Process,r,dumb1,dumb2);
  disp(' ');
  disp(' ');
  disp('Enter the name of the RED TARGET to be used (".tiff" assumed)');
  Targetname=input(':: ','s');

[Fusedtarget,Tm1,Tn1,t_r,dumb1,dumb2]=get_file(Fusedtarget,Targetname,TProcess,t_r,dumb1,dumb2);
  disp(' ');
  disp(' ');
  disp('Enter the RED TARGET TEMPLATE filename (".tiff" assumed)');
  Tempname=input(':: ','s');

[dumb3,Tm1,Tn1,Tr_template,dumb1,dumb2]=get_file(dumb3,Tempname,TProcess,Tr_template,dumb1,
dumb2);
 end;

 if keep_g=='Y'
  disp(' ');
  disp(' ');
  disp('Enter the name of the GREEN BACKGROUND image to be used (".tiff") assumed');
  Imagename=input(':: ','s');
  [FusedImage,m,n,g,dumb1,dumb2]=get_file(FusedImage,Imagename,Process,g,dumb1,dumb2);
  disp(' ');
  disp(' ');
  disp('Enter the name of the GREEN TARGET to be used (".tiff" assumed)');
  Targetname=input(':: ','s');

[Fusedtarget,Tm1,Tn1,t_g,dumb1,dumb2]=get_file(Fusedtarget,Targetname,TProcess,t_g,dumb1,dumb2);
  disp(' ');
  disp(' ');
  disp('Enter the GREEN TARGET TEMPLATE filename (".tiff" assumed)');
  Tempname=input(':: ','s');

[dumb3,Tm1,Tn1,Tg_template,dumb1,dumb2]=get_file(dumb3,Tempname,TProcess,Tg_template,dumb1,
dumb2);
 end;
```

```
 if keep_b=='Y'
  disp(' ');
  disp(' ');
  disp('Enter the name of the BLUE BACKGROUND image to be used (".tiff") assumed');
  Imagename=input(':: ','s');
  [FusedImage,m,n,b,dumb1,dumb2]=get_file(FusedImage,Imagename,Process,b,dumb1,dumb2);
  disp(' ');
  disp(' ');
  disp('Enter the name of the BLUE TARGET to be used (".tiff" assumed)');
  Targetname=input(':: ','s');

[Fusedtarget,Tm1,Tn1,t_b,dumb1,dumb2]=get_file(Fusedtarget,Targetname,TProcess,t_b,dumb1,dumb2);
  disp(' ');
  disp(' ');
  disp('Enter the BLUE TARGET TEMPLATE filename (".tiff" assumed)');
  Tempname=input(':: ','s');

[dumb3,Tm1,Tn1,Tb_template,dumb1,dumb2]=get_file(dumb3,Tempname,TProcess,Tb_template,dumb1,
dumb2);
 end;

 Imagename=FusedImage;
 Targetname=Fusedtarget;
 clear FusedImage Fusedtarget TProcess dumb1 dumb2 dumb3 Tempname;

else
 if Process==1
  keep_r='Y';
  keep_g='N';
  keep_b='N';
  keep_C='N';
 else
  keep_r='Y';
  keep_g='Y';
  keep_b='Y';
  keep_C='Y';
 end;
 TProcess=1; dumb1=0; dumb2=0; dumb3=[]; m=0; n=0; r=[]; g=[]; b=[];
 Tm1=0; Tn1=0; t_r=[]; t_g=[]; t_b=[]; Tr_template=[]; Tg_template=[]; Tb_template=[];
 disp(' ');
 disp(' ');
 disp('Enter the name of the BACKGROUND image to be used (".tiff") assumed');
 Imagename=input(':: ','s');
 [dumb3,m,n,r,g,b]=get_file(dumb3,Imagename,Process,r,g,b);
 disp(' ');
 disp(' ');
 disp('Enter the name of the TARGET to be used (".tiff" assumed)');
 Targetname=input(':: ','s');
 [dumb3,Tm1,Tn1,t_r,t_g,t_b]=get_file(dumb3,Targetname,Process,t_r,t_g,t_b);
 disp(' ');
 disp(' ');
 disp('Enter the TARGET TEMPLATE filename (".tiff" assumed)');
 Tempname=input(':: ','s');
 TProcess=1;
```

```
[dumb3,Tm1,Tn1,Tr_template,dumb1,dumb2]=get_file(dumb3,Tempname,TProcess,Tr_template,dumb1,
dumb2);
 Tg_template=Tr_template;
 Tb_template=Tr_template;
 clear dumb1 dumb2 dumb3 TProcess
end;
```

M-file to choose color components.

filename: **choose_components.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%
%    This MATLAB m-file is a sub-program of "go_MCfilter". This routine
% allows specification of the number of components to be included in the
% development and evaluation of two options.  In the case of processing
% option [2], it specifies which bands are to be fused together such as
% in the case of fusing up to three different bands (colors).  In the
% case of processing option [3], it specifies the actual RGB color bands
% which are to be evaluated when one of more of the bands are redundant.
%
%
function [keep_r, keep_g, keep_b] = choose_components (r_equals_g,r_equals_b,g_equals_b,msg);

 if msg=='Y'
  disp(' ');
  disp(' ');
  disp('              ********************************');
  disp('       *********  REDUNDANT COMPONENTS EXIST  *********');
  disp('              ********************************');
  disp(' ');
  if r_equals_g == 0
   disp(' RED   component identical to the GREEN component...');
  elseif r_equals_b ==0
   disp(' RED   component identical to the BLUE  component...');
  elseif g_equals_b == 0
   disp(' GREEN component identical to the BLUE  component...');
  end;
 end;
 Pick='*';
 while Pick=='*'
  disp(' ');
  disp(' Choices: ');
  disp('         [S]  SPECIFY the components to continue with');
  disp('         [A]  continue with ALL THREE components');
  disp(' ');
  Pick =  input(' Enter an "S" or "A" ::  ','s');
  if Pick~='S' & Pick~='A'
   Pick='*';
  end;
 end;
 if Pick == 'S'
  keep_r='*';
  keep_g='*';
  keep_b='*';
  disp(' ');
```

```
 disp('Enter "Y" to include the component in the analysis, or "N" to exclude...');
 disp(' ');
 while keep_r~='Y' & keep_r~='N'
  keep_r = input('RED ?  ("Y" or "N") : ','s');
 end;
 while keep_g~='Y' & keep_g~='N'
  keep_g = input('GREEN ? ("Y" or "N") : ','s');
 end;
 while keep_b~='Y' & keep_b~='N'
  keep_b = input('BLUE ?  ("Y" or "N") : ','s');
 end;
end;
```

M-file to load tiff images.

Filename: **get_file.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%
%    This MATLAB m-file is a sub-program of "go_MCfilter".  This routine
% is used to read in the TIFF formatted image files.  There are to formats
% of TIFF images.  If Tifftype==8 then the file has been stored in the
% "indexed" format.  If Tifftype==24 then the file has been stored in the
% "RGB" format.  Each requires different loading assignments and are
% distinguished automatically and read in appropriately.
%
%
function [Fusedname,mdim,ndim,comp1,comp2,comp3] = get_file (Fusedname, Name,
Process,comp1,comp2,comp3)

filename=[Name,'.tiff'];
Fusedname=[Fusedname,Name];
TiffType=tiffread1(filename);
if TiffType==8
  [x,map]=tiffread1(filename);
  % RE-FORMAT BACKGROUND DATA.
  if Process==3
    disp('Re-formating INDEXED Background data into RGB...')
    [comp1,comp2,comp3]=ind2rgb(x,map);
  else
    comp1=ind2gray(x,map);
    comp2=[];
    comp3=[];
    % display using imshow1(comp1,64) command
  end;
  [mdim,ndim]=size(comp1);
elseif TiffType==24
  [comp1,comp2,comp3]=tiffread1(filename);
  if Process==1
    comp1=rgb2gray(comp1,comp2,comp3);
    comp2=[];
    comp3=[];
    % display using imshow1(r,64) command
  end;
  [mdim,ndim]=size(comp1);
else
  disp('This file does NOT conform to INDEXED or RGB as expected...');
end;
```

APPENDIX C. M-FILE TO INPUT FILTERING OPTIONS

Filename: **get_filt_info.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
%
%    This MATLAB m-file is a sub-program of "go_MCfilter".  This is just a
% a series of questions in order to get the filtering information and
% preprocessing options.
%
%

% Query whether background image and/or/nor target are to be Normalized...
BNtype='*';
TNtype='*';
while BNtype~='U' & BNtype~='N' & TNtype~='U' & TNtype~='N'
 disp(' ');
 disp(' ');
 BNtype=input('Leave the BACKGROUND image [U]n-normalized or [N]ormalize: ','s');
 disp(' ');
 disp(' ');
 TNtype=input('Leave the TARGET image [U]n-normalized or [N]ormalize: ', 's');
end;


% Query whether this will be using a Local or Global Matched Filter or a
% Template matching filter.
Filtertype='*';
while Filtertype~='L' & Filtertype~='G' & Filtertype~='T'
 disp(' ');
 disp(' ');
 disp(' ');
 disp('Matched Filter processing using: ');
 disp(' ');
 disp('     [ L ]  Local Matched Filter');
 disp('              (Developed using the statistics of a specified field or point within BKGD).');
 disp('     [ G ]  Global Matched Filter');
 disp('              (Developed using the statistics of the entire BKGD image).');
 disp('     [ T ]  Target "as is" as Filter');
 disp(' ');
 disp(' ');
 disp(' ');
 Filtertype=input('Enter an "L", "G" or a "T" : ', 's');
end;

% Determine target and filter dimensions
nn=round((n/2)-0.01);
mm=round((m/2)-0.01);
Tm2=round((Tm1-1)/2);
Tn2=round((Tn1-1)/2);
if Filtertype=='T'
```

```
  Tm3=Tm2;
  Tn3=Tn2;
 else
  Tm3=(Tm2 + round(sqrt(Tm1)));
  Tn3=(Tn2 + round(sqrt(Tn1)));
 end;
 Tn4=(2*Tn3+1);
 Tm4=(2*Tm3+1);

 % Placement of a target in a specified field or point within the background image
 Choice='*';
 while Choice~='F' & Choice~='P'
  disp(' ');
  disp(' ');
  disp('Target placement:');
  disp(' ');
  disp('    [F] Evaluation of target at intervals within a specified FIELD');
  disp('    [P] Evaluation of target at a single specified POINT (with prev. calc. coefs.)');
  disp(' ');
  Choice=input('Choose [F] or [P]: ','s');
 end;

% Initialize range variables and show the target's dimentions so that the user can choose a horizontal and
vertical step size.
Xmin=0;
Xmax=0;
Ymin=0;
Ymax=0;
disp(' ');
disp(' ');
disp(['background image dimensions are: (height x width) = (',int2str(m),' x ',int2str(n),')']);
disp(' ');
disp(['Target filter dimensions are : (height x width) = (',int2str(Tm1),' x ',int2str(Tn1),')']);



if Choice=='F'   % if [F]ield is specified, the get the corner points of the
            % specified field and adjust the background as necessary.

  disp(' ');
  disp(' ');
  skipX = input('Enter the HORIZONTAL interval for target insertion into the field:  ');
  disp(' ');
  skipY =  input('Enter the VERTICAL interval for target insertion into the field:  ');
  disp(' ');
  disp('  Placement of target(s) in a specified field with specified intervals');
  disp(' ');
  disp('Enter the (x,y)-coords of the upperleft and lower right corners of the field:');
  while Xmin<Tn3+1 | Xmin>(n-Tn3) | Xmin>Xmax | Xmax<Tn3+1 | Xmax>(n-Tn3) | Ymin<Tm3+1 |
Ymin>(m-Tm3) | Ymin>Ymax | Ymax<Tm3+1 | Ymax>(m-Tm3)
   disp(' ');
   disp('***NOTE: X values must be between:');
   disp(' Min      Max');
```

```
    disp([Tn3+1, n-Tn3]);
    disp('***NOTE: Y values must be between:');
    disp(' Min      Max');
    disp([Tm3+1, m-Tm3]);
    Xmin = input('  Enter the upper left X-coords of field : ');
    Ymin = input('  Enter the upper left Y-coords of field : ');
    Xmax = input('  Enter the lower right X-coords of field as: ');
    Ymax = input('  Enter the lower right Y-coords of field as: ');
    disp(' ');
    disp(' ');
    FSeries='1';
    FSeries=input('Which Field in this series (ie: "1", "2", "3",...? : ','s');
    Foptions=[int2str(Process),BNtype,TNtype,Filtertype,Choice,FSeries];
  end;
  on=n;
  om=m;
  oXmin=Xmin;
  oYmin=Ymin;
  oXmax=Xmax;
  oYmax=Ymax;
  m_offset=0;
  n_offset=0;
% if a local filter has been designated then the background is truncated to the region of concern
  if (Filtertype=='L' | Filtertype=='T')
    % set-up for local bkgd mean and stdev for each component.
    if keep_r=='Y'
      r=r(Ymin-Tm3:Ymax+Tm3,Xmin-Tn3:Xmax+Tn3);
      [m,n]=size(r);
    end;
    if keep_g=='Y'
      g=g(Ymin-Tm3:Ymax+Tm3,Xmin-Tn3:Xmax+Tn3);
      [m,n]=size(g);
    end;
    if keep_b=='Y'
      b=b(Ymin-Tm3:Ymax+Tm3,Xmin-Tn3:Xmax+Tn3);
      [m,n]=size(b);
    end;
%   set the m & n offsets so that the original pixel row and col indices are kept intact
    m_offset=Ymin-Tm3-1;
    n_offset=Xmin-Tn3-1;
    nn=round((n/2)-0.01);
    mm=round((m/2)-0.01);
    Xmin=Tn3+1;
    Xmax=n-Tn3;
    Ymin=Tm3+1;
    Ymax=m-Tm3;
  end;



else %    if [P]oint is specified then use the image/target/options information
    % to determine if the [F]ield option for the same image/target/options
    % has been run since it will use the same filter coefficients used when
    % when the point within the specified field where evaluated.
```

```
disp(' ');
disp(' ');
disp('Point evaluations require the use of previously calculated');
disp('filter coefficients calculated from a field and stored as: ');
Foptions=[int2str(Process),BNtype,TNtype,Filtertype,'F'];
disp([Imagename,'_',Targetname,'_',Foptions,'?']);
disp(' ');
FSeries=input('Which Field serial number (ie: "1", "2", "3",...)? : ','s');
disp(' ');
disp(' ');
disp('Checking for previously calculated Filter coefficients...');
disp(' ');
Foptions=[Foptions,FSeries];
Coeffile=[Imagename,'_',Targetname,'_',Foptions,'.coef'];
[fid, message]=fopen(Coeffile,'r','b');
if fid==-1
 disp('No luck... ');
 disp(' ');
 disp('You MUST have run the Field Option to calculated filter coefficients...');
 disp(' ');
 disp('Hit ctrl-c and restart program...');
 stop;
else
 disp('Loading filter coefficients from:');
 disp(Coeffile);
 Ilength=fread(fid,[1,2],'short');
 Sourcename=setstr(fread(fid,Ilength,'char'));
 Tlength=fread(fid,[1,2],'short');
 Tgtused=setstr(fread(fid,Tlength,'char'));
 Flength=fread(fid,[1,2],'short');
 Finfo=fread(fid,Flength,'char');
 keep_r=setstr(fread(fid,[1,1],'char'));
 keep_g=setstr(fread(fid,[1,1],'char'));
 keep_b=setstr(fread(fid,[1,1],'char'));
 keep_C=setstr(fread(fid,[1,1],'char'));
 om=fread(fid,[1,1],'short');
 on=fread(fid,[1,1],'short');
 oXmin=fread(fid,[1,1],'short');
 oYmin=fread(fid,[1,1],'short');
 oXmax=fread(fid,[1,1],'short');
 oYmax=fread(fid,[1,1],'short');
 h_rgb_dim=fread(fid,[1,2],'short');
 if keep_C=='Y'
  h_123_dim=fread(fid,[1,2],'short');
 end;
 if keep_r=='Y'
  h_r=fread(fid,h_rgb_dim,'float64');
  if keep_C=='Y'
   h1=fread(fid,h_123_dim,'float64');
  end;
  r_mean=fread(fid,[1,1],'float64');
```

```
    r_std=fread(fid,[1,1],'float64');
   end;
   if keep_g=='Y'
    h_g=fread(fid,h_rgb_dim,'float64');
    if keep_C=='Y'
     h2=fread(fid,h_123_dim,'float64');
    end;
    g_mean=fread(fid,[1,1],'float64');
    g_std=fread(fid,[1,1],'float64');
   end;
   if keep_b=='Y'
    h_b=fread(fid,h_rgb_dim,'float64');
    if keep_C=='Y'
     h3=fread(fid,h_123_dim,'float64');
    end;
    b_mean=fread(fid,[1,1],'float64');
    b_std=fread(fid,[1,1],'float64');
   end;
   fclose(fid);
   disp('These Coefficients created using :');
   disp([Sourcename,' with targets ',Tgtused,' and filter options (',Finfo,')']);
end;

while (Xmin<oXmin) | (Xmin>oXmax) | (Ymin<oYmin) | (Ymin>oYmax)
  disp(' ');
  disp(' ');
  skipX=1;
  skipY=1;
  disp(' ');
  disp('***NOTE: X values must be between:');
  disp(' Min      Max');
  disp([Tn3+1, n-Tn3]);
  disp('***NOTE: Y values must be between:');
  disp(' Min      Max');
  disp([Tm3+1, m-Tm3]);
  Xmin=input('  Enter the x-coord for placing the target: ');
  Xmax=Xmin;
  Ymin=input('  Enter the y-coord for placing the target: ');
  Ymax=Ymin;
  disp(' ');
  disp(' ');
  PSeries='1';
  PSeries=input('Which POINT in this series (ie: "1", "2", "3",...)? : ','s');
  Foptions=[int2str(Process),BNtype,TNtype,Filtertype,'P',PSeries];
end;
if keep_r=='Y'
  r=r(Ymin-Tm3:Ymax+Tm3,Xmin-Tn3:Xmax+Tn3);
  [m,n]=size(r);
end;
if keep_g=='Y'
  g=g(Ymin-Tm3:Ymax+Tm3,Xmin-Tn3:Xmax+Tn3);
  [m,n]=size(g);
end;
if keep_b=='Y'
```

```
    b=b(Ymin-Tm3:Ymax+Tm3,Xmin-Tn3:Xmax+Tn3);
    [m,n]=size(b);
  end;
  m_offset=Ymin-Tm3-1;
  n_offset=Xmin-Tn3-1;
  nn=Tn3+1;
  mm=Tm3+1;
  Xmin=Tn3+1;
  Xmax=Tn3+1;
  Ymin=Tm3+1;
  Ymax=Tm3+1;
end;
```

APPENDIX D. M-FILE TO PREPROCESS IMAGES AND/OR TARGETS

Filename: **do_preprocessing.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%    This MATLAB m-file is a sub-program of "go_MCfilter".  This code does
% preprocessing if specified in the get_filter_info routine.  The previously
% specified preprocessing choices are stored in the variables BNtype and
% TNtype.  The preprocessing is normalization.
%
%

bkgd_r=r;
bkgd_g=g;
bkgd_b=b;

% Normalize Background and/or/nor the Target as specified
if BNtype=='N' | TNtype=='N'
 if Choice=='F'
  if keep_r=='Y'
   r_mean=mean(r(:));
   r_std=std(r(:));
  end;
  if keep_g=='Y'
   g_mean=mean(g(:));
   g_std=std(g(:));
  end;
  if keep_b=='Y'
   b_mean=mean(b(:));
   b_std=std(b(:));
  end;
 end;

% If specified, Normalize the Background first
 if BNtype=='N'
  bkgd_r=[];
  bkgd_g=[];
  bkgd_b=[];
  if keep_r=='Y'
   UniformCheck=abs(r - r_mean);
   if (mean(UniformCheck(:))) >= .01
    bkgd_r =(r - r_mean);
    if r_std~=0
     bkgd_r=bkgd_r/r_std;
    end;
   else
    bkgd_r=r;
    disp('bkgd_r=r...');
   end;
```

```
    end;
    if keep_g=='Y'
     UniformCheck=abs( g - g_mean);
     if (mean(UniformCheck(:))) >= .01
      bkgd_g = ( g - g_mean);
      if g_std~=0
       bkgd_g=bkgd_g/g_std;
      end;
     else
      bkgd_g=g;
      disp('bkgd_g=g...');
     end;
    end;
    if keep_b=='Y'
     UniformCheck=abs( b - b_mean);
     if (mean(UniformCheck(:))) >= .01
      bkgd_b = ( b - b_mean);
      if b_std~=0
       bkgd_b=bkgd_b/b_std;
      end;
     else
      bkgd_b=b;
      disp('bkgd_b=b...');
     end;
    end;
  end;

% If Specified, Normalize the Target next.
  if TNtype=='N'
   if keep_r=='Y'
    tr=t_r-r_mean;
    if r_std~=0
     t_r=tr/r_std;
    end;
    tr=t_r .* (Tr_template==0);
    t_r=tr;
    clear tr;
   end;
   if keep_g=='Y'
    tg=t_g-g_mean;
    if g_std~=0
     t_g=tg/g_std;
    end;
    tg=t_g .* (Tg_template==0);
    t_g=tg;
    clear tg;
   end;
   if keep_b=='Y'
    tb=t_b-b_mean;
    if b_std~=0
     t_b=tb/b_std;
    end;
    tb=t_b .* (Tb_template==0);
    t_b=tb;
```

```
    clear tb
   end;
  end;

else  %  If not specified then assign standard values to be saved.
  r_mean=0;
  r_std=1;
  g_mean=0;
  g_std=1;
  b_mean=0;
  b_std=1;
end;

clear r g b UniformCheck
```

APPENDIX E. M-FILE TO DEVELOP MATCHED FILTER COEFICIENTS

Filename: **make_GLfilters.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%
%     This M-file computes both 2-D spatial 3-D matched filter coefficients
% (spatial-spectral) using the input background data and target model.
%
%
% Uses: (other m-files)
%       choose_components.m
%       create_2D.m
%       create_3d.m


start_time = clock;

if Process==3
  r_equals_g = nnz(bkgd_r - bkgd_g) ~= 0;  % 0 if red component equals green (1 otherwise)
  r_equals_b = nnz(bkgd_r - bkgd_b) ~= 0;  % 0 if red component equals blue (1 otherwise)
  g_equals_b = nnz(bkgd_g - bkgd_b) ~= 0;  % 0 if green component equals blue (1 otherwise)
  redundant = r_equals_g + r_equals_b + g_equals_b;
  keep_r = 'Y';
  keep_g = 'Y';
  keep_b = 'Y';
  keep_C = 'Y';
  if redundant < 3
     msg='Y';
    [keep_r,keep_g,keep_b]=choose_components(r_equals_g,r_equals_b,g_equals_b,msg);
  end;
elseif Process==1
  keep_r = 'Y';
  keep_g = 'N';
  keep_b = 'N';
  keep_C = 'N';
end;

% PRODUCE 2-D HANNING WINDOW FOR USE IN BLURRING AND SMOOTHING
disp(' ');
disp(' ');
disp('Checking for previously calculated hanning window to match bkgd dimensions...');
disp(' ');
Hanfile=[int2str(m),'x',int2str(n),'.han'];
[fid, message]=fopen(Hanfile,'r','b');
if fid==-1
  disp(' ');
  disp('No luck... must create new window. ');
  disp(' ');
```

```
   disp('Creating new Hanning Window...');
   arraydimx = n;
   arraydimy = m;
   for k=1:m;
    for l=1:n;
     length = (  (k-(arraydimy/2+.5))^2 + (l-(arraydimx/2+.5))^2  )^0.5;
          if  length > arraydimy/2
            han(k,l) = 0;
          else
            han(k,l) = ( cos( (pi/(1.0*m)) * length) )^2;
          end
    end
   end
   fid=fopen(Hanfile,'w','b');
   fwrite(fid,han,'float64');
   fclose(fid);
 else
   disp('LOADING 2-D HANNING WINDOW');
   han=fread(fid,[m,n],'float64');
   fclose(fid);
 end;

 target_r=[];
 target_g=[];
 target_b=[];

 if keep_r=='Y'
   target_r=zeros(m,n);
   target_r((mm-Tm2):(mm+Tm2),(nn-Tn2):(nn+Tn2)) = t_r;
 end

 if keep_g=='Y'
   target_g=zeros(m,n);
   target_g((mm-Tm2):(mm+Tm2),(nn-Tn2):(nn+Tn2)) = t_g;
 end

 if keep_b=='Y'
   target_b=zeros(m,n);
   target_b((mm-Tm2):(mm+Tm2),(nn-Tn2):(nn+Tn2)) = t_b;
 end

 % CALCULATE 2-D MATCHED FILTER COEFFICIENTS FOR EACH BAND
   disp('CALCULATING 2-D MATCHED FILTER COEFFICIENTS FOR EACH BAND')

 h_r = [];
 h_g = [];
 h_b = [];

 if keep_r == 'Y'
   disp('start red:');
   disp(clock);
   h_r=create_2D(target_r, bkgd_r, han, mm, nn, Tm3, Tn3);
   h_rgb_dim=size(h_r);
 end;
```

```
if keep_g == 'Y'
  disp('start green:');
  disp(clock);
  h_g=create_2D(target_g, bkgd_g, han, mm, nn, Tm3, Tn3);
  h_rgb_dim=size(h_g);
end;

if keep_b == 'Y'
  disp('start blue:');
  disp(clock);
  h_b=create_2D(target_b, bkgd_b, han, mm, nn, Tm3, Tn3);
  h_rgb_dim=size(h_b);
end;

% CALCULATE 3-D MATCHED FILTER COEFFICIENTS
if keep_C=='Y'
  disp('CALCULATING 3-D MATCHED FILTER COEFFICIENTS')
  [h1,h2,h3,h_123_dim]=create_3d(target_r, target_g, target_b, han, keep_r, keep_g, keep_b, bkgd_r,
bkgd_g, bkgd_b, mm, nn, Tm3, Tn3);
end;

Coeffile=[Imagename,'_',Targetname,'_',Foptions,'.coef'];
fid=fopen(Coeffile,'w','b');
Ilength=size(Imagename);
fwrite(fid,Ilength,'short');
fwrite(fid,Imagename,'char');
Tlength=size(Targetname);
fwrite(fid,Tlength,'short');
fwrite(fid,Targetname,'char');
Flength=size(Foptions);
fwrite(fid,Flength,'short');
fwrite(fid,Foptions,'char');
fwrite(fid,keep_r,'char');
fwrite(fid,keep_g,'char');
fwrite(fid,keep_b,'char');
fwrite(fid,keep_C,'char');
fwrite(fid,om,'short');
fwrite(fid,on,'short');
fwrite(fid,oXmin,'short');
fwrite(fid,oYmin,'short');
fwrite(fid,oXmax,'short');
fwrite(fid,oYmax,'short');
fwrite(fid,h_rgb_dim,'short');
if keep_C=='Y'
  fwrite(fid,h_123_dim,'short');
end;
if keep_r=='Y'
  fwrite(fid,h_r,'float64');
  if keep_C=='Y'
    fwrite(fid,h1,'float64');
  end;
  fwrite(fid,r_mean,'float64');
  fwrite(fid,r_std,'float64');
```

```
 end;
 if keep_g=='Y'
  fwrite(fid,h_g,'float64');
  if keep_C=='Y'
    fwrite(fid,h2,'float64');
  end;
  fwrite(fid,g_mean,'float64');
  fwrite(fid,g_std,'float64');
 end;
 if keep_b=='Y'
  fwrite(fid,h_b,'float64');
  if keep_C=='Y'
    fwrite(fid,h3,'float64');
  end;
  fwrite(fid,b_mean,'float64');
  fwrite(fid,b_std,'float64');
 end;
 fclose(fid);

 disp(start_time);
 end_time = clock
```

M-file deriving 2-D matched filter coefficients.

Filename: **create_2D.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%
%      This M-file function creates a 2-D matched filter which is slightly
% larger than the original target (increased by twice the square-root of
% the dimensions of the original target.
%
%

[h_2D] = create_2D (target, bkgd, han, mm, nn, Tm3, Tn3)
S = fft2(target); % converts the spatial target into spectral domain.
West = abs(fft2(bkgd)).^2; % produces the spectral power of the background noise.
Rest = ifft2(West); % converts the background spectral power back into the spatial domain.
R = Rest .* (fftshift(han)); % smoothes and averages the background noise.
W = fft2(R);   % converts "smoothed" background noise back into the spectral domain.
H = W.\conj(S); % convolves the inverted "smoothed" background noise with the target.
h_2D = real(ifft2(H)); % converts this result into the spatial domain.
h_2D = h_2D((mm+1-Tm3):(mm+1+Tm3),(nn+1-Tn3):(nn+1+Tn3)); % crops filter to size.
```

M-file deriving 3-D matched filter coefficients.

Filename: **create_3d.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%
%     This M-file function creates a 3-D matched filter which is slightly
% larger than the original target (increased by twice the square-root of
% the dimensions of the original target.  Same technique as used in Create_2D.m
% but carried out on the third (Spectral) dimension.
%
% uses:
%     fft3_9(...)
%     ifft3_9(...)
%
%

function [h1,h2,h3,h_123_dim] = create_3d (target_r, target_g, target_b, han, keep_r, keep_g, keep_b,
bkgd_r, bkgd_g, bkgd_b, mm, nn, Tm3, Tn3)

 disp('start target fft3: (step 1 of 5)');
 disp(clock);

 [S1,S2,S3]=fft3_9(target_r,target_g,target_b,keep_r,keep_g,keep_b);
                          clear target_r target_g target_b;
 disp('start bkgd fft3: (step 2 of 5)');
 disp(clock);
 [BKGD_1,BKGD_2,BKGD_3]=fft3_9(bkgd_r,bkgd_g,bkgd_b,keep_r,keep_g,keep_b);

 West1=[];
 West2=[];
 West3=[];
 if keep_r=='Y'
   West1=abs(BKGD_1).^2;
 end;
 if keep_g=='Y'
   West2=abs(BKGD_2).^2;
 end;
 if keep_b=='Y'
   West3=abs(BKGD_3).^2;
 end;
                                          clear BKGD_1 BKGD_2 BKGD_3;
 disp('start REST123 ifft3: (step 3 of 5)');
 disp(clock);
 [Rest1,Rest2,Rest3]=ifft3_9(West1,West2,West3,keep_r,keep_g,keep_b);
                                          clear West1 West2 West3;
 R1=[];
 R2=[];
 R3=[];
```

```
if keep_r=='Y'
 R1=Rest1.*(fftshift(han))*1;
end;
if keep_g=='Y'
 R2=Rest2.*(fftshift(han))*0;
end;
if keep_b=='Y'
 R3=Rest3.*(fftshift(han))*0;
end;
                                        clear Rest1 Rest2 Rest3;
disp('start W123 fft3: (step 4 of 5)');
disp(clock);
[W1,W2,W3]=fft3_9(R1,R2,R3,keep_r,keep_g,keep_b);
                                        clear R1 R2 R3;
H1=[];
H2=[];
H3=[];
if keep_r=='Y'
 H1=W1.\conj(S1);
end;
if keep_g=='Y' & keep_b=='Y'
 H3=W2.\conj(S2);       % interchange of H2 and H3 compensates for 3D conj()
 H2=W3.\conj(S3);
elseif keep_g=='Y' & keep_b=='N'
 H2=W2.\conj(S2);
elseif keep_g=='N' & keep_b=='Y'
 H3=W3.\conj(S3);
end;
                                        clear W1 W2 W3 S1 S2 S3;
h1=[];
h2=[];
h3=[];
disp('start h123 ifft3: (step 5 of 5)');
disp(clock);
[h1,h2,h3] = ifft3_9(H1,H2,H3,keep_r,keep_g,keep_b);

                                        clear H1 H2 H3;
if keep_r=='Y'
 h1 = real( h1((mm+1-Tm3):(mm+1+Tm3),(nn+1-Tn3):(nn+1+Tn3)) );
 h_123_dim=size(h1);
end;
if keep_g=='Y'
 h2 = real( h2((mm+1-Tm3):(mm+1+Tm3),(nn+1-Tn3):(nn+1+Tn3)) );
 h_123_dim=size(h2);
end;
if keep_b=='Y'
 h3 = real( h3((mm+1-Tm3):(mm+1+Tm3),(nn+1-Tn3):(nn+1+Tn3)) );
 h_123_dim=size(h3);
end;
```

APPENDIX F. M-FILE TO DEVELOP A TEMPLATE MATCHING FILTER

Filename: **make_Tfilters.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
%
%
%   This M-file simply defines the necessary filters for Template
% matching by using the targets themselves and "reversing" them for
% the filters.
%
%
% Uses: (other m-files)
%       choose_components.m
%       reverse.m
%
%

start_time = clock;
if keep_C=='Y'
 if Process==3
   r_equals_g = nnz(bkgd_r - bkgd_g) ~= 0;  % 0 if red component equals green (1 otherwise)
   r_equals_b = nnz(bkgd_r - bkgd_b) ~= 0;  % 0 if red component equals blue (1 otherwise)
   g_equals_b = nnz(bkgd_g - bkgd_b) ~= 0;  % 0 if green component equals blue (1 otherwise)
   redundant = r_equals_g + r_equals_b + g_equals_b;
   keep_r = 'Y';
   keep_g = 'N';
   keep_b = 'Y';
   keep_C = 'Y';
   if redundant < 3
     msg='Y';
     [keep_r,keep_g,keep_b]=choose_components(r_equals_g,r_equals_b,g_equals_b,msg);
   end;
 end;
 if keep_r=='Y'
   h_r= reverse(t_r);
   h_rgb_dim=size(h_r);
   if keep_C=='Y'
     h1=h_r;
     h_123_dim=size(h1);
   end;
 end;
 if keep_g=='Y'
   h_g= reverse(t_g);
   h_rgb_dim=size(h_g);
   if keep_C=='Y'
     h2=h_g;
     h_123_dim=size(h2);
   end;
 end;
 if keep_b=='Y'
```

```
    h_b= reverse(t_b);
    h_rgb_dim=size(h_b);
    if keep_C=='Y'
     h3=h_b;
     h_123_dim=size(h3);
    end;
  end;
 else
  keep_r = 'Y';
  keep_g = 'N';
  keep_b = 'N';
  keep_C = 'N';
  h_r= reverse(t_r);
  h_rgb_dim=size(h_r);
  h_template=(h_r==0);
 end;

 Coeffile=[Imagename,'_',Targetname,'_',Foptions,'.coef'];
 fid=fopen(Coeffile,'w','b');
 Ilength=size(Imagename);
 fwrite(fid,Ilength,'short');
 fwrite(fid,Imagename,'char');
 Tlength=size(Targetname);
 fwrite(fid,Tlength,'short');
 fwrite(fid,Targetname,'char');
 Flength=size(Foptions);
 fwrite(fid,Flength,'short');
 fwrite(fid,Foptions,'char');
 fwrite(fid,keep_r,'char');
 fwrite(fid,keep_g,'char');
 fwrite(fid,keep_b,'char');
 fwrite(fid,keep_C,'char');
 fwrite(fid,om,'short');
 fwrite(fid,on,'short');
 fwrite(fid,oXmin,'short');
 fwrite(fid,oYmin,'short');
 fwrite(fid,oXmax,'short');
 fwrite(fid,oYmax,'short');
 fwrite(fid,h_rgb_dim,'short');
 if keep_C=='Y'
  fwrite(fid,h_123_dim,'short');
 end;
 if keep_r=='Y'
  fwrite(fid,h_r,'float64');
  if keep_C=='Y'
   fwrite(fid,h1,'float64');
  end;
  fwrite(fid,r_mean,'float64');
  fwrite(fid,r_std,'float64');
 end;
 if keep_g=='Y'
  fwrite(fid,h_g,'float64');
  if keep_C=='Y'
   fwrite(fid,h2,'float64');
```

```
  end;
  fwrite(fid,g_mean,'float64');
  fwrite(fid,g_std,'float64');
end;
if keep_b=='Y'
  fwrite(fid,h_b,'float64');
  if keep_C=='Y'
    fwrite(fid,h3,'float64');
  end;
  fwrite(fid,b_mean,'float64');
  fwrite(fid,b_std,'float64');
end;
fclose(fid);
disp(start_time);
end_time = clock
```

M-file to invert the target template.

Filename: **reverse.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%
%   This M-file simply defines the necessary filters for Template
% matching by using the targets themselves and "reversing" them for
% the filters.
%
%
function [RevFilter] = reverse (Filter)
 RevFilter=Filter;
 [Ymax,Xmax]=size(Filter);
 for i=1:Ymax
  for j=1:Xmax
   RevFilter((Ymax+1)-i,(Xmax+1)-j)=Filter(i,j);
  end;
 end;
```

APPENDIX G. M-FILE TO SEQUENTIALLY FILTER THE IMAGE

Filename: **filter_image.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%    This MATLAB m-file is a sub-program of "go_MCfilter".
%
% This M-file uses the matched filter coefficients (both 2-D spatial and 3-D
% spatio-spectral) and performs sequential filtering operations by starting
% from the top of the specified field (or the point coordinates) and evaluates
% one point by convoluting the respective filter with the respective target
% inserted at that point to get the "signal+noise" filter output value.
% In order to get the "noise" output value of the specified background, the
% respective filter is convolved with the entire background of interest and
% stored.  Thus, the noise for any specific point within the image is
% efficiently calculated and paired to the "signal+noise" output value
% corresponding to the point being evaluated.
%
% Uses: (other m-files)
%       filter_with_2D.m
%       filter_with_3D.m
%

time1=clock;


disp('Running matched filter across image field... ');
disp('Started at time: ');
clock

% BEGIN DO-LOOPS WHICH CREATE SMALL IMAGE SEGMENTS AND PERFORM FILTERING
clear summary summarys summaryn
summary = []; summarys = []; summaryn = [];     % null matrix for later storing results
scene_r=[]; scene_g=[]; scene_b=[];
if keep_r=='Y'
 disp('red component setup...')
 clock
 noise_r=conv2(bkgd_r,h_r,'same');
 if keep_C=='Y'
   disp('bonus 3rd-dim red component setup...');
   clock
   noise_1=conv2(bkgd_r,h1,'same');
 end;
end;
if keep_g=='Y'
 disp('green component setup...');
 clock
 noise_g=conv2(bkgd_g,h_g,'same');
 if keep_C=='Y'
```

```
    disp('bonus 3rd-dim green component setup...');
    clock
    noise_2=conv2(bkgd_g,h2,'same');
  end;
end;
if keep_b=='Y'
  disp('blue component setup...');
  clock
  noise_b=conv2(bkgd_b,h_b,'same');
  if keep_C=='Y'
    disp('bonus 3rd-dim blue component setup...');
    clock
    noise_3=conv2(bkgd_b,h3,'same');
  end;
end;

ipos=[];
jpos=[];
Num=0;
for i = Ymin:skipY:Ymax          % locations of targets in y-direction
  disp(i+m_offset)
  disp(clock)
  for j = Xmin:skipX:Xmax        % locations of targets in x-direction
    ipos=[ipos; i+m_offset];
    jpos=[jpos; j+n_offset];
    Num=Num+1;
% FILTER LWIR SCENE (RED) USING 2-D SPATIAL FILTER
    fil_SNR_2dr = 0;
    fil_NNR_2dr = 0;
    if keep_r == 'Y'
%     disp('FILTERING LWIR SCENE (RED) USING 2-D SPATIAL FILTER')
      fil_NNR_2dr=noise_r(i,j);
      [fil_SNR_2dr, scene_r] = filter_with_2D (i,j, bkgd_r, h_r, t_r, mm, nn, Tm2, Tn2, Tm3, Tn3,
Tr_template);
    end;


% FILTER MWIR SCENE (GREEN) USING 2-D SPATIAL FILTER
    fil_SNR_2dg = 0;
    fil_NNR_2dg = 0;
    if keep_g == 'Y'
%     disp('FILTERING MWIR SCENE (GREEN) USING 2-D SPATIAL FILTER')
      fil_NNR_2dg=noise_g(i,j);
      [fil_SNR_2dg, scene_g] = filter_with_2D (i,j, bkgd_g, h_g, t_g, mm, nn, Tm2, Tn2, Tm3, Tn3,
Tg_template);
    end;


% FILTER SWIR SCENE (BLUE) USING 2-D SPATIAL FILTER
    fil_SNR_2db = 0;
    fil_NNR_2db = 0;
    if keep_b == 'Y'
%     disp('FILTERING SWIR SCENE (BLUE) USING 2-D SPATIAL FILTER')
      fil_NNR_2db=noise_b(i,j);
```

```
    [fil_SNR_2db, scene_b] = filter_with_2D (i,j, bkgd_b, h_b, t_b, mm, nn, Tm2, Tn2, Tm3, Tn3,
Tb_template);
  end;


% FILTER COMPOSITE SCENE (COLOR) USING 3-D SPATIAL FILTER
  fil_SNR_3d=0;
  fil_NNR_3d=0;
  if keep_C=='Y'
%   disp('FILTERING COMPOSITE SCENE (COLOR) USING 3-D SPATIAL FILTER')
    fil_NNR_3d=0;
    if keep_r=='Y'
      fil_NNR_3d=fil_NNR_3d+noise_r(i,j);
    end;
    if keep_g=='Y'
      fil_NNR_3d=fil_NNR_3d+noise_g(i,j);
    end;
    if keep_b=='Y'
      fil_NNR_3d=fil_NNR_3d+noise_b(i,j);
    end;
%   [fil_SNR_3d] = filter_with_3D (scene_r, scene_b, scene_g, keep_r, h1, h2, h3, Tm3, Tn3);
    [Hm Hn]=size(h_123_dim);
    output1=zeros(Hm,Hn);
    output2=zeros(Hm,Hn);
    output3=zeros(Hm,Hn);
    if keep_r=='Y'
%     disp('make output1...');
     output1 = conv2(h1,scene_r,'same');          % filtering
     fil_SNR_3d=fil_SNR_3d+output1(Tm3+1,Tn3+1);          % pt. of filter peak resonse - max signal
    end;
    if keep_g=='Y'
%     disp('make output2...');
     output2 = conv2(h2,scene_g,'same');          % filtering
     fil_SNR_3d=fil_SNR_3d+output2(Tm3+1,Tn3+1);          % pt. of filter peak resonse - max signal
    end;
    if keep_b=='Y'
%     disp('make output3...');
     output3 = conv2(h3,scene_b,'same');          % filtering
     fil_SNR_3d=fil_SNR_3d+output3(Tm3+1,Tn3+1);          % pt. of filter peak resonse - max signal
    end;
                              clear output1 output2 output3
  end;


% ADD SUMMARY DATA FOR CURRENT IMAGE SEGMENT
  summarys = [summarys ; fil_SNR_2dr fil_SNR_2dg fil_SNR_2db fil_SNR_3d];
  summaryn = [summaryn ; fil_NNR_2dr fil_NNR_2dg fil_NNR_2db fil_NNR_3d];
  index = 1:1:Num;
  summary = [index' summarys summaryn ipos jpos];
 end;
end;
summary2=summary;
save fdat8.mat summary2;
Sum_dim=size(summary);
```

```
Datafile=[Imagename,'_',Targetname,'_',Foptions,'.dat'];
fid=fopen(Datafile,'w','b');
Ilength=size(Imagename);
fwrite(fid,Ilength,'short');
fwrite(fid,Imagename,'char');
Tlength=size(Targetname);
fwrite(fid,Tlength,'short');
fwrite(fid,Targetname,'char');
Flength=size(Foptions);
fwrite(fid,Flength,'short');
fwrite(fid,Foptions,'char');
fwrite(fid,keep_r,'char');
fwrite(fid,keep_g,'char');
fwrite(fid,keep_b,'char');
fwrite(fid,keep_C,'char');
fwrite(fid,om,'short');
fwrite(fid,on,'short');
fwrite(fid,oXmin,'short');
fwrite(fid,oYmin,'short');
fwrite(fid,oXmax,'short');
fwrite(fid,oYmax,'short');
fwrite(fid,skipX,'short');
fwrite(fid,skipY,'short');
fwrite(fid,Sum_dim,'short');
fwrite(fid,summary,'float64');
fclose(fid);

disp('Start time:');
disp(time1);
time2=clock
```

M-file to filter with the 2-D filter at location (i,j).

Filename: **filter_with_2D.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%    This MATLAB m-file is a sub-program of "go_MCfilter".  This code uses the
% given template to "cutout" the area in which the target is to be added to using
% element by element multiplication and then matrix addition for "inserting" the
% target into the area "cutout" for it.
%
%
function [fil_SNR_2d, scene] = filter_with_2D (i, j, bkgd, h_2D, target, mm, nn, Tm2, Tn2, Tm3, Tn3,
T_plate)

scene = bkgd( (i-Tm3):(i+Tm3) , (j-Tn3):(j+Tn3) );
T_scene= scene((Tm3+1-Tm2):(Tm3+1+Tm2),(Tn3+1-Tn2):(Tn3+1+Tn2));
T_scene=(T_scene .* T_plate) + target;
scene((Tm3+1-Tm2):(Tm3+1+Tm2),(Tn3+1-Tn2):(Tn3+1+Tn2))=T_scene;
output = conv2(h_2D,scene,'same');     % filtering
fil_sig_max =  output(Tm3+1,Tn3+1);   % pt. of filter peak resonse - max signal
fil_SNR_2d = fil_sig_max;
```

M-file to filter with the 3-D filters at location(i,j).

Filename: **filter_with_3D.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%    This MATLAB m-file is a sub-program of "go_MCfilter".  This code uses the
% given template to "cutout" the area in which the target is to be added to using
% element by element multiplication and then matrix addition for "inserting" the
% target into the area "cutout" for it. Done for up to three specified components.
%
%
function [fil_SNR_3d] = filter_with_3D (scene_r, scene_b, scene_g, keep_r, keep_g, keep_b, h1, h2, h3,
Tm3, Tn3)

 [Hm Hn]=size(h1);
 Tm3
 output1=zeros(Hm,Hn);
 output2=zeros(Hm,Hn);
 output3=zeros(Hm,Hn);
 fil_sig_max=0;
 if keep_r=='Y'
%  disp('make output1...');
   output1 = conv2(h1,scene_r,'same');        % filtering
   fil_sig_max=fil_sig_max+output1(Tm3+1,Tn3+1);        % pt. of filter peak resonse - max signal
 end;
 if keep_g=='Y'
%  disp('make output2...');
   output2 = conv2(h2,scene_g,'same');        % filtering
   fil_sig_max=fil_sig_max+output2(Tm3+1,Tn3+1);        % pt. of filter peak resonse - max signal
 end;
 if keep_b=='Y'
%  disp('make output3...');
   output3 = conv2(h3,scene_b,'same');        % filtering
   fil_sig_max=fil_sig_max+output3(Tm3+1,Tn3+1);        % pt. of filter peak resonse - max signal
 end;
                              clear output1 output2 output3
 fil_SNR_3d = fil_sig_max;                  % calc. SNR
```

APPENDIX H. M-FILE TO FILTER SEVERAL POSITIONS AT ONCE.

Filename: **conv_image.m**

```
%
% SUBPROGRAM: MATLAB 5.0 m-files
% Written by: James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
%
%   This MATLAB m-file is a sub-program of "go_MCfilter".
%
%   This MATLAB m-file does the same filtering as the "filter_image"
% routine but can make use of convoluting the entire background with several
% targets inserted at several of the points to be evaluated.  This alternate
% method could save time if enough of the targets can be inserted into the
% specified field and probably only in the case of the trunctated background.
%
% Uses: (other m-files)
%       make_COandADDIN.m
%

if keep_r=='Y'
 disp('red component setup...')
 clock
 Noise_r=conv2(bkgd_r,h_r,'same');
 if keep_C=='Y'
  disp('bonus 3rd-dim red component setup...');
  clock
  Noise_1=conv2(bkgd_r,h1,'same');
 end;
end;
if keep_g=='Y'
 disp('green component setup...');
 clock
 Noise_g=conv2(bkgd_g,h_g,'same');
 if keep_C=='Y'
  disp('bonus 3rd-dim green component setup...');
  clock
  Noise_2=conv2(bkgd_g,h2,'same');
 end;
end;
if keep_b=='Y'
 disp('blue component setup...');
 clock
 Noise_b=conv2(bkgd_b,h_b,'same');
 if keep_C=='Y'
  disp('bonus 3rd-dim blue component setup...');
  clock
  Noise_3=conv2(bkgd_b,h3,'same');
 end;
end;
fil_SNR_2dr=[];
fil_SNR_2dg=[];
fil_SNR_2db=[];
```

```
fil_SNR_3d=[];
fil_NNR_2dr=[];
fil_NNR_2dg=[];
fil_NNR_2db=[];
fil_NNR_3d=[];

Yshift=skipY;
if skipY>Tm4
  Yshift=skipY;
end;
Xshift=skipX;
if skipX>Tn4
  Xshift=skipX;
end;
NeedNewCOandADDIN='Y';

ipos=[];
jpos=[];
Num=0;
for inum=Ymin:Yshift:(Ymin+Tm4)
 for jnum=Xmin:Xshift:(Xmin+Tn4)
   disp(['Upper left target--> (',int2str(inum),',',int2str(jnum),')']);
   clock
   if NeedNewCOandADDIN=='Y'
    disp('Creating new Cutouts and Addins...');
    clock
    if keep_r=='Y'
      [Cutout_r,Cm,Cn,Addin_r,X_pad,Y_pad]=make_COandADDIN
(Tm1,Tn1,Tm2,Tn2,Tm4,Tn4,Tr_template,t_r,inum,jnum,Xmax,Ymax,skipX,skipY,m,n);
    end;
    if keep_g=='Y'
      [Cutout_g,Cm,Cn,Addin_g,X_pad,Y_pad]=make_COandADDIN
(Tm1,Tn1,Tm2,Tn2,Tm4,Tn4,Tg_template,t_g,inum,jnum,Xmax,Ymax,skipX,skipY,m,n);
    end;
    if keep_b=='Y'
      [Cutout_b,Cm,Cn,Addin_b,X_pad,Y_pad]=make_COandADDIN
(Tm1,Tn1,Tm2,Tn2,Tm4,Tn4,Tb_template,t_b,inum,jnum,Xmax,Ymax,skipX,skipY,m,n);
    end;
    NeedNewCOandADDIN='N';
  end;

  if ((inum-Tm2+Cm)>m) | ((jnum-Tn2+Cn)>n)
    NeedNewCOandADDIN='Y';
    disp('Need new Cutouts and Addins...');
  else
    NeedNewCOandADDIN='N';
    disp('Go with current Cutouts and Addins...');
    Top=(inum-Tm2);
    Bottom=(inum-Tm2+Cm-1);
    Left=(jnum-Tn2);
    Right=(jnum-Tn2+Cn-1);
  end;
  if NeedNewCOandADDIN=='N'
    if keep_r=='Y'
```

```
   disp('Convolving Targets in RED component (2D)...');
   clock;
   Scene_r=bkgd_r;
   Scene_r(Top:Bottom,Left:Right)=Scene_r(Top:Bottom,Left:Right) .* Cutout_r + Addin_r;
   SN_r=conv2(Scene_r,h_r,'same');
   if keep_C=='Y'
    disp('Convolving Targets in RED component (3D)...');
    clock;
    SN_1=conv2(Scene_r,h1,'same');
   end;
  end;
  if keep_g=='Y'
   disp('Convolving Targets in GREEN component (2D)...');
   clock;
   Scene_g=bkgd_g;
   Scene_g(Top:Bottom,Left:Right)=Scene_g(Top:Bottom,Left:Right) .* Cutout_g + Addin_g;
   SN_g=conv2(Scene_g,h_g,'same');
   if keep_C=='Y'
    disp('Convolving Targets in GREEN component (3D)...');
    clock;
    SN_2=conv2(Scene_g,h2,'same');
   end;
  end;
  if keep_b=='Y'
   disp('Convolving Targets in BLUE component (2D)...');
   clock;
   Scene_b=bkgd_b;
   Scene_b(Top:Bottom,Left:Right)=Scene_b(Top:Bottom,Left:Right) .* Cutout_b + Addin_b;
   SN_b=conv2(Scene_b,h_b,'same');
   if keep_C=='Y'
    disp('Convolving Targets in BLUE component (3D)...');
    clock;
    SN_3=conv2(Scene_b,h3,'same');
   end;
  end;
end;

disp('Extracting results...');
clock

for inum2=inum:Y_pad:Ymax
 for jnum2=jnum:X_pad:Xmax
   ipos=[ipos; inum2+m_offset];
   jpos=[jpos; jnum2+n_offset];
   Num=Num+1;
   if keep_r=='Y'
    fil_SNR_2dr=[fil_SNR_2dr; SN_r(inum2,jnum2)];
    fil_NNR_2dr=[fil_NNR_2dr; Noise_r(inum2,jnum2)];
   else
    fil_SNR_2dr=[fil_SNR_2dr; 0];
    fil_NNR_2dr=[fil_NNR_2dr; 0];
   end;
   if keep_g=='Y'
    fil_SNR_2dg=[fil_SNR_2dg; SN_g(inum2,jnum2)];
```

```
        fil_NNR_2dg=[fil_NNR_2dg; Noise_g(inum2,jnum2)];
       else
        fil_SNR_2dg=[fil_SNR_2dg; 0];
        fil_NNR_2dg=[fil_NNR_2dg; 0];
       end;
       if keep_b=='Y'
        fil_SNR_2db=[fil_SNR_2db; SN_b(inum2,jnum2)];
        fil_NNR_2db=[fil_NNR_2db; Noise_b(inum2,jnum2)];
       else
        fil_SNR_2db=[fil_SNR_2db; 0];
        fil_NNR_2db=[fil_NNR_2db; 0];
       end;
       if keep_C=='Y'
        fil_sig_max=0;
        fil_null_max=0;
        if keep_r=='Y'
         fil_sig_max=fil_sig_max+SN_1(inum2,jnum2);
         fil_null_max=fil_null_max+Noise_1(inum2,jnum2);
        end;
        if keep_g=='Y'
         fil_sig_max=fil_sig_max+SN_2(inum2,jnum2);
         fil_null_max=fil_null_max+Noise_2(inum2,jnum2)
        end;
        if keep_b=='Y'
         fil_sig_max=fil_sig_max+SN_3(inum2,jnum2);
         fil_null_max=fil_null_max+Noise_3(inum2,jnum2);
        end;
        fil_SNR_3d=[fil_SNR_3d; fil_sig_max];
        fil_NNR_3d=[fil_NNR_3d; fil_null_max];
       else
        fil_SNR_2dr=[fil_SNR_2dr; 0];
        fil_NNR_2dr=[fil_NNR_2dr; 0];
       end;
     end;
   end;
   disp('done extracting results...');
   clock
  end;
end;
index=1:1:Num;
summaryS=[fil_SNR_2dr,fil_SNR_2dg,fil_SNR_2db,fil_SNR_3d];
summaryN=[fil_NNR_2dr,fil_NNR_2dg,fil_NNR_2db,fil_NNR_3d];
summary=[index' summaryS summaryN ipos jpos];
summary2=summary;
save fdat8.mat summary2;
Sum_dim=size(summary);

Datafile=[Imagename,'_',Targetname,'_',Foptions,'.dat'];
fid=fopen(Datafile,'w','b');
Ilength=size(Imagename);
fwrite(fid,Ilength,'short');
fwrite(fid,Imagename,'char');
Tlength=size(Targetname);
fwrite(fid,Tlength,'short');
```

```
fwrite(fid,Targetname,'char');
Flength=size(Foptions);
fwrite(fid,Flength,'short');
fwrite(fid,Foptions,'char');
fwrite(fid,keep_r,'char');
fwrite(fid,keep_g,'char');
fwrite(fid,keep_b,'char');
fwrite(fid,keep_C,'char');
fwrite(fid,om,'short');
fwrite(fid,on,'short');
fwrite(fid,oXmin,'short');
fwrite(fid,oYmin,'short');
fwrite(fid,oXmax,'short');
fwrite(fid,oYmax,'short');
fwrite(fid,skipX,'short');
fwrite(fid,skipY,'short');
fwrite(fid,Sum_dim,'short');
fwrite(fid,summary,'float64');
fclose(fid);

disp('Finished...');
clock
```

M-file to create cutout and add-in patterns.

Filename: **make_COandADDIN.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
%
%    This MATLAB m-file is a sub-program of "go_MCfilter".  This code creates
% a "pattern" of target cutouts and a corresponding "pattern" of targets which
% is used in the same manner of a "single" cutout and add done in filter_with_2D
% or 3D m-files.  However, this "pattern" cuts-out and adds-in several targets
% at once.
%
%
function [Cutout,Cm,Cn,Addin,X_pad,Y_pad] = make_COandADDIN
(Tm1,Tn1,Tm2,Tn2,Tm4,Tn4,Tplate,Tgt,inum,jnum,Xmax,Ymax,SkipX,SkipY,m,n)

Cutout=[];
COline=[];
Addin=[];
AIline=[];
Y_pad=Tm4;
if SkipY>Tm4
  Y_pad=SkipY;
end;
Y_COspacer=ones(Y_pad-Tm1,Tn1);
Y_ADDINspacer=zeros(Y_pad-Tm1,Tn1);
X_pad=Tn4;
if SkipX>Tn4
  X_pad=SkipX;
end;
for inum2=inum:Y_pad:Ymax
  if (inum2+Tm2)<m
    COline=[COline; Tplate];
    AIline=[AIline; Tgt];
    if (inum2+Tm2+Y_pad)<m
      COline=[COline; Y_COspacer];
      AIline=[AIline; Y_ADDINspacer];
    end;
  end;
end;
[COm, COn]=size(COline);
X_COspacer=ones(COm,X_pad-Tn1);
X_ADDINspacer=zeros(COm,X_pad-Tn1);
for jnum2=jnum:X_pad:Xmax
  if (jnum2+Tn2)<n
    Cutout=[Cutout,COline];
    Addin=[Addin,AIline];
    if (jnum2+Tn2+X_pad)<n
      Cutout=[Cutout, X_COspacer];
      Addin=[Addin, X_ADDINspacer];
    end;
```

```
  end;
end;
[Cm,Cn]=size(Cutout);
```

APPENDIX I. M-FILE TO DISPLAY RESULTS AS A ROC PLOT

Filename: **roc9.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%   This MATLAB m-file is a sub-program of "go_MCfilter".
%
%       This M-file takes the output results from the multicolor matched filter
% processing (both 2-D spatial and 3-D spatio-spectral) and computes data for
% the ROC plots.  This information is then output to the screen and saved to a
% file using the Imagename_Targetname_OPTIONS.roc filename which can be viewed
% at a later time using "recall_roc9".
%
% uses:
%     Threshold.m
%
%

[rows,cols] = size(summary);
count = rows;

sig_r = summary(:,2);
sig_g = summary(:,3);
sig_b = summary(:,4);
sig_c = summary(:,5);

null_r = summary(:,6);
null_g = summary(:,7);
null_b = summary(:,8);
null_c = summary(:,9);
Fused=[];
result_r=0;
if keep_r=='Y'
 if mean(sig_r(:))<mean(null_r(:))
  sig_r=-sig_r;
  null_r=-null_r;
  disp(' ');
  disp(' ');
  STRmsg='NOTE!: "inverse" energy situation- data inverted';
  if keep_C=='Y'
   STRmsg=[STRmsg,' for RED component'];
   Fused=[Fused,'red '];
  end;
  disp(STRmsg);
 end;
 % COMPUTE ROC-CURVE FOR LWIR (RED) TARGET AND BACKGROUND
 disp('COMPUTING ROC-CURVE FOR LWIR (RED) TARGET AND BACKGROUND')
 result_r = Threshold(sig_r, null_r, count);
 res_size=size(result_r);
```

```
 end;
                                        clear result threshold sig null pmr far
 result_g=0;
 if keep_g=='Y'
  if mean(sig_g(:))<mean(null_g(:))
    sig_g=-sig_g;
    null_g=-null_g;
    disp(' ');
    disp(' ');
    STRmsg='NOTE!: "inverse" energy situation- data inverted';
    if keep_C=='Y'
      STRmsg=[STRmsg,' for GREEN component'];
      Fused=[Fused,'green '];
    end;
    disp(STRmsg);
  end;
  % COMPUTE ROC-CURVE FOR MWIR (GREEN) TARGET AND BACKGROUND
  disp('COMPUTING ROC-CURVE FOR MWIR (GREEN) TARGET AND BACKGROUND')
  result_g = Threshold(sig_g, null_g, count);
  res_size=size(result_g);
end;
                                        clear result threshold sig null pmr far
result_b=0;
if keep_b=='Y'
  if mean(sig_b(:))<mean(null_b(:))
    sig_b=-sig_b;
    null_b=-null_b;
    disp(' ');
    disp(' ');
    STRmsg='NOTE!: "inverse" energy situation- data inverted';
    if keep_C=='Y'
      STRmsg=[STRmsg,' for BLUE component'];
      Fused=[Fused,'blue '];
    end;
    disp(STRmsg);
  end;
  % COMPUTE ROC-CURVE FOR SWIR (BLUE) TARGET AND BACKGROUND
  disp('COMPUTING ROC-CURVE FOR SWIR (BLUE) TARGET AND BACKGROUND')
  result_b = Threshold(sig_b, null_b, count);
  res_size=size(result_b);
end;
                                        clear result threshold sig null pmr far
result_c=0;
if keep_C=='Y'
  if mean(sig_c(:))<mean(null_c(:))
    sig_c=-sig_c;
    null_c=-null_c;
    disp(' ');
    disp(' ');
    STRmsg='NOTE!: "inverse" energy situation- data inverted';
    if keep_C=='Y'
      STRmsg=[STRmsg,' for FUSED(',Fused,') '];
    end;
    disp(STRmsg);
```

```
 end;
 % COMPUTE ROC-CURVE FOR COMPOSITE (COLOR) TARGET AND BACKGROUND
 disp('COMPUTING ROC-CURVE FOR COMPOSITE (COLOR) TARGET AND BACKGROUND')
 result_c = Threshold(sig_c, null_c, count);
 res_size=size(result_c);
end;
                                clear result threshold sig null pmr far

% PLOT RESULTS USING MATLAB GRAPHICS
disp('PLOT RESULTS USING MATLAB GRAPHICS')
figure;
if keep_C=='Y' & size(result_c)>[1,1]
 plot(result_c(:,1),result_c(:,2),'y');
 hold on;
end;
if keep_r=='Y' & size(result_r)>[1,1]
 plot(result_r(:,1),result_r(:,2),'r:');
 hold on;
end;
if keep_g=='Y' & size(result_g)>[1,1]
 plot(result_g(:,1),result_g(:,2),'g:');
 hold on;
end;
if keep_b=='Y' & size(result_b)>[1,1]
 plot(result_b(:,1),result_b(:,2),'b:');
 hold on;
end;
ROCtitle=['Empirical ROC for ',Imagename,' with ',Targetname,' (',Foptions,')'];
title(ROCtitle);
ylabel('probability of detection');
xlabel('false alarm rate');
axis([-.025,1.025,-0.025,1.025])
hold off;

Resultsfile=[Imagename,'_',Targetname,'_',Foptions,'.roc'];
fid=fopen(Resultsfile,'w','b');
Ilength=size(Imagename);
fwrite(fid,Ilength,'short');
fwrite(fid,Imagename,'char');
Tlength=size(Targetname);
fwrite(fid,Tlength,'short');
fwrite(fid,Targetname,'char');
Flength=size(Foptions);
fwrite(fid,Flength,'short');
fwrite(fid,Foptions,'char');
fwrite(fid,keep_r,'char');
fwrite(fid,keep_g,'char');
fwrite(fid,keep_b,'char');
fwrite(fid,keep_C,'char');
fwrite(fid,res_size,'short');
if keep_r=='Y'
 fwrite(fid,result_r,'float64');
end;
if keep_g=='Y'
```

```
  fwrite(fid,result_g,'float64');
end;
if keep_b=='Y'
  fwrite(fid,result_b,'float64');
end;
if keep_C=='Y'
  fwrite(fid,result_c,'float64');
end;
fclose(fid);
```

M-file to vary the threshold criteria.

Filename: **Threshold.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%    This MATLAB m-file is a sub-program of "go_MCfilter".  This code varies
% the threshold criteria from the maximum value down to the minimum value
% (100 decriments) and calculates the associated Pd and Pfa at each threshold
% criteria.
%
%
function [result] = Threshold (sig,null,count)
 result=[];
 maximum = max( [sig; null] );
 minimum = min( [sig; null] );
 if maximum <= minimum
  disp('PROBLEM OF MAX CLUTTER LEAKAGE ALWAYS < MIN SIGNAL');
 end
 increment = (maximum - minimum) / 100;
 for threshold = (maximum+2*increment):-increment:minimum
  s_exceed = 0;
  n_exceed = 0;
  s_exceed = sig >= threshold;
  s_exceed = sum(s_exceed);
  n_exceed = null >= threshold;
  n_exceed = sum(n_exceed);
  pmd = 1-((count - s_exceed) / count);      % probability of detection
  far = n_exceed / count;                    % false alarm rate
  result = [result ; far pmd];
 end
```

APPENDIX J. M-FILES TO DISPLAY DATA

Filename: **recall_roc9.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%    This MATLAB m-file is a sub-program of "go_MCfilter".  This code allows
% you to redisplay any previously evaluated image_target_OPTIONS ROC plot by
% recalling the stored information in the associated ".roc" file.  No
% recalculations are needed.
%
%
fid=-1;
while fid==-1
  disp(' ');
  disp(' ');
  Results=input('Enter file with roc results (".roc" assumed): ','s');
  Resultsfile=[Results,'.roc'];
  fid=fopen(Resultsfile,'r');
end;
Ilength=fread(fid,[1,2],'short');
Imagename=fread(fid,Ilength,'char');
Imagename=setstr(Imagename);
Tlength=fread(fid,[1,2],'short');
Targetname=fread(fid,Tlength,'char');
Targetname=setstr(Targetname);
Flength=fread(fid,[1,2],'short');
Foptions=fread(fid,Flength,'char');
Foptions=setstr(Foptions);
keep_r=setstr(fread(fid,[1,1],'char'));
keep_g=setstr(fread(fid,[1,1],'char'));
keep_b=setstr(fread(fid,[1,1],'char'));
keep_C=setstr(fread(fid,[1,1],'char'));
res_size=fread(fid,[1,2],'short');
if keep_r=='Y'
  result_r=fread(fid,res_size,'float64');
end;
if keep_g=='Y'
  result_g=fread(fid,res_size,'float64');
end;
if keep_b=='Y'
  result_b=fread(fid,res_size,'float64');
end;
if keep_C=='Y'
  result_c=fread(fid,res_size,'float64');
end;
fclose(fid);

% PLOT RESULTS USING MATLAB GRAPHICS
  disp('PLOT RESULTS USING MATLAB GRAPHICS')
```

```
figure;
if keep_C=='Y' & size(result_c)>[1,1]
  plot(result_c(:,1),result_c(:,2),'y');
  hold on;
end;
if keep_r=='Y' & size(result_r)>[1,1]
  plot(result_r(:,1),result_r(:,2),'r:');
  hold on;
end;
if keep_g=='Y' & size(result_g)>[1,1]
  plot(result_g(:,1),result_g(:,2),'g:');
  hold on;
end;
if keep_b=='Y' & size(result_b)>[1,1]
  plot(result_b(:,1),result_b(:,2),'b:');
  hold on;
end;
ROCtitle=['Empirical ROC for ',Imagename,' with ',Targetname,' (',Foptions,')'];
title(ROCtitle);
ylabel('probability of detection');
xlabel('false alarm rate');
axis([-.025,1.025,-0.025,1.025])
hold off;
```

M-file to show the 2-D and 3-D filters.

Filename: **show_coefs.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%    This MATLAB m-file is a sub-program of "go_MCfilter".  This allows you to
% graphically inspect the calculated 2-D and 3-D filters for any evaluated
% image_target_OPTIONS.coef file.
%
%
fid=-1;
while fid==-1
 disp(' ');
 disp(' ');
 Coef=input('Enter the Coeficients FILE (".coef" assumed): ','s');
 Coeffile=[Coef,'.coef'];
 fid=fopen(Coeffile,'r');
end;
disp('Loading 2-D and 3-D matched filter coefficients from:');
disp(Coeffile);
Ilength=fread(fid,[1,2],'short');
Imagename=fread(fid,Ilength,'char');
Imagename=setstr(Imagename);
Tlength=fread(fid,[1,2],'short');
Targetname=fread(fid,Tlength,'char');
Targetname=setstr(Targetname);
Flength=fread(fid,[1,2],'short');
Foptions=fread(fid,Flength,'char');
Foptions=setstr(Foptions);
keep_r=setstr(fread(fid,[1,1],'char'));
keep_g=setstr(fread(fid,[1,1],'char'));
keep_b=setstr(fread(fid,[1,1],'char'));
keep_C=setstr(fread(fid,[1,1],'char'));
m=fread(fid,[1,1],'short');
n=fread(fid,[1,1],'short');
Xmin=fread(fid,[1,1],'short');
Ymin=fread(fid,[1,1],'short');
Xmax=fread(fid,[1,1],'short');
Ymax=fread(fid,[1,1],'short');
h_rgb_dim=fread(fid,[1,2],'short');
if keep_C=='Y'
 h_123_dim=fread(fid,[1,2],'short');
end;
if keep_r=='Y'
 h_r=fread(fid,h_rgb_dim,'float64');
 if keep_C=='Y'
  h1=fread(fid,h_123_dim,'float64');
 end;
 r_mean=fread(fid,[1,1],'float64');
```

```
    r_std=fread(fid,[1,1],'float64');
   end;
   if keep_g=='Y'
    h_g=fread(fid,h_rgb_dim,'float64');
    if keep_C=='Y'
     h2=fread(fid,h_123_dim,'float64');
    end;
    g_mean=fread(fid,[1,1],'float64');
    g_std=fread(fid,[1,1],'float64');
   end;
   if keep_b=='Y'
    h_b=fread(fid,h_rgb_dim,'float64');
    if keep_C=='Y'
     h3=fread(fid,h_123_dim,'float64');
    end;
    b_mean=fread(fid,[1,1],'float64');
    b_std=fread(fid,[1,1],'float64');
   end;
   fclose(fid);
   SourceInfo=[Imagename,' with ',Targetname,' (',Foptions,')'];
   if keep_r=='Y'
    figure;
    imshow1(h_r);
    STRtitle=['2D filter for ',SourceInfo];
    if keep_C=='Y'
     STRtitle=['RED(2D) filter for ',SourceInfo];
    end;
    title(STRtitle);
    colormap(gray);
    if keep_C=='Y'
     figure;
     imshow1(h1);
     STRtitle=['RED(3D) filter for ',SourceInfo];
     title(STRtitle);
     colormap(gray);
    end;
   end;
   if keep_g=='Y'
    figure;
    imshow1(h_g);
    STRtitle=['2D filter for ',SourceInfo];
    if keep_C=='Y'
     STRtitle=['GREEN(2D) filter for ',SourceInfo];
    end;
    title(STRtitle);
    colormap(gray);
    if keep_C=='Y'
     figure;
     imshow1(h2);
     STRtitle=['GREEN(3D) filter for ',SourceInfo];
     title(STRtitle);
     colormap(gray);
    end;
   end;
```

```
if keep_b=='Y'
 figure;
 imshow1(h_b);
 STRtitle=['2D filter for ',SourceInfo];
 if keep_C=='Y'
  STRtitle=['BLUE(2D) filter for ',SourceInfo];
 end;
 title(STRtitle);
 colormap(gray);
 if keep_C=='Y'
  figure;
  imshow1(h3);
  STRtitle=['BLUE(3D) filter for ',SourceInfo];
  title(STRtitle);
  colormap(gray);
 end;
end;
```

M-file to graphically display the data.

Filename: **display_data.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%    This MATLAB m-file is a sub-program of "go_MCfilter".  This allows you to
% graphically inspect the data produced by the filter evaluation.  It simply
% recalls the actual data stored in the image_targer_OPTIONS.dat file and allows
% you to see the histograms and/or the actual field of values for any specified
% components that apply.
%
%
disp(' ');
disp(' ');
Data=input('Enter name of the Datafile (".dat" assumed): ','s');
disp(' ');
disp(' ');
ShowDataPoints=input('Show data "field" information with histograms? (Y/N) : ','s');
Datafile=[Data,'.dat'];
Dfid=fopen(Datafile,'r');
Ilength=fread(Dfid,[1,2],'short');
Imagename=fread(Dfid,Ilength,'char');
Imagename=setstr(Imagename);
Tlength=fread(Dfid,[1,2],'short');
Targetname=fread(Dfid,Tlength,'char');
Targetname=setstr(Targetname);
Flength=fread(Dfid,[1,2],'short');
Foptions=fread(Dfid,Flength,'char');
Foptions=setstr(Foptions);
keep_r=setstr(fread(Dfid,[1,1],'char'));
keep_g=setstr(fread(Dfid,[1,1],'char'));
keep_b=setstr(fread(Dfid,[1,1],'char'));
keep_C=setstr(fread(Dfid,[1,1],'char'));
om=fread(Dfid,[1,1],'short');
on=fread(Dfid,[1,1],'short');
oXmin=fread(Dfid,[1,1],'short');
oYmin=fread(Dfid,[1,1],'short');
oXmax=fread(Dfid,[1,1],'short');
oYmax=fread(Dfid,[1,1],'short');
skipX=fread(Dfid,[1,1],'short');
skipY=fread(Dfid,[1,1],'short');
Sum_dim=fread(Dfid,[1,2],'short');
Fsummary=fread(Dfid,Sum_dim,'float64');
fclose(Dfid);
disp(' ');
disp(' ');
disp('Data loaded for the following components:');
show_r=keep_r;
show_g=keep_g;
```

```
show_b=keep_b;
show_C=keep_C;
if keep_r=='Y'
  disp('Red ');
  show_r='*';
end;
if keep_g=='Y'
  disp('Green ');
  show_g='*';
end;
if keep_b=='Y'
  disp('Blue ');
  show_b='*';
end;
if keep_C=='Y'
  disp('Color ');
  show_C='*';
end;
disp(' ');
disp(' ');
disp('Specify which data to show:');
disp(' ');
disp(' ');
while show_r~='Y' & show_r~='N'
  show_r = input('RED ?  ("Y" or "N") : ','s');
end;
while show_g~='Y' & show_g~='N'
  show_g = input('GREEN ? ("Y" or "N") : ','s');
end;
while show_b~='Y' & show_b~='N'
  show_b = input('BLUE ?  ("Y" or "N") : ','s');
end;
while show_C~='Y' & show_C~='N'
  show_C = input('COLOR ?  ("Y" or "N") : ','s');
end;


SourceInfo=[Imagename,' with ',Targetname,'(',Foptions,')'];
sig_r = Fsummary(:,2);
sig_g = Fsummary(:,3);
sig_b = Fsummary(:,4);
sig_c = Fsummary(:,5);
null_r = Fsummary(:,6);
null_g = Fsummary(:,7);
null_b = Fsummary(:,8);
null_c = Fsummary(:,9);
RowIndex=Fsummary(:,10);
ColIndex=Fsummary(:,11);
Fused=[];
if show_r=='Y'
  maxX=abs(max([sig_r;null_r]));
  E=ceil(log10(maxX));
  Estr=int2str(E);
  Etxt=[];
```

```
lEstr=length(Estr);
for Epos=1:1:lEstr
  Etxt=[Etxt,'^',Estr(Epos)];
end;
sig_r=sig_r/(10^E);
null_r=null_r/(10^E);
if mean(sig_r)<mean(null_r)
  sig_r=-sig_r;
  null_r=-null_r;
  disp(' ');
  disp(' ');
  STRmsg='NOTE!: "inverse" energy situation- data inverted';
  if keep_C=='Y'
    STRmsg=[STRmsg,' for RED component'];
  end;
  disp(STRmsg);
end;
if ShowDataPoints=='Y'
  SubOptNoise_r=sparse(RowIndex,ColIndex,null_r,om,on);
  SubOptSignal_r=sparse(RowIndex,ColIndex,sig_r,om,on);
end;
maxX=max([sig_r;null_r]);
minX=min([sig_r;null_r]);
Inc=(maxX-minX)/50;
N=minX:Inc:maxX;
figure
hist([null_r,sig_r],N)
STRtitle=SourceInfo;
STRxlabel=['2D Filter Output (x10',Etxt,')'];
if keep_C=='Y'
  STRtitle=[STRtitle,' (RED component)'];
  STRxlabel=['RED component ',STRxlabel];
  Fused=[Fused,' red'];
end;
title(STRtitle);
xlabel(STRxlabel);
ylabel('Number of Occurrences');
legend('Noise','Sig+Noise',0);
if ShowDataPoints=='Y'
 Zmin=min([null_r; sig_r; 0]);
 if Zmin<0
   Tplate=SubOptNoise_r==0;
   Tplate=Tplate*Zmin;
   SubOptNoise_r=SubOptNoise_r+Tplate;
   Tplate=SubOptSignal_r==0;
   Tplate=Tplate*Zmin;
   SubOptSignal_r=SubOptSignal_r+Tplate;
   clear Tplate
 end;
 Zmax=max([null_r; sig_r; 0]);
 figure
 mesh(SubOptNoise_r)
 axis([1 on 1 om Zmin Zmax])
 axis ij
```

```
    title(['Noise for ',STRtitle]);
    ylabel('Image pixel row number');
    xlabel('Image pixel column number');
    zlabel(STRxlabel);
    figure
    mesh(SubOptSignal_r)
    axis([1 on 1 om Zmin Zmax])
    axis ij
    title(['Signal+Noise for ',STRtitle]);
    ylabel('Image pixel row number');
    xlabel('Image pixel column number');
    zlabel(STRxlabel);
  end;
end;
if show_g=='Y'
  maxX=abs(max([sig_g;null_g]));
  E=ceil(log10(maxX));
  Estr=int2str(E);
  Etxt=[];
  lEstr=length(Estr);
  for Epos=1:1:lEstr
    Etxt=[Etxt,'^',Estr(Epos)];
  end;
  sig_g=sig_g/(10^E);
  null_g=null_g/(10^E);
  if mean(sig_g)<mean(null_g)
    sig_g=-sig_g;
    null_g=-null_g;
    disp(' ');
    disp(' ');
    STRmsg='NOTE!: "inverse" energy situation- data inverted';
    if keep_C=='Y'
      STRmsg=[STRmsg,' for GREEN component'];
    end;
    disp(STRmsg);
  end;
  if ShowDataPoints=='Y'
    SubOptNoise_g=sparse(RowIndex,ColIndex,null_g,om,on);
    SubOptSignal_g=sparse(RowIndex,ColIndex,sig_g,om,on);
  end;
  Top=max([sig_g;null_g]);
  Bott=min([sig_g;null_g]);
  Inc=(Top-Bott)/50;
  N=Bott:Inc:Top;
  figure
  hist([null_g,sig_g],N)
  STRtitle=SourceInfo;
  STRxlabel=['2D Matched Filter Output (x10',Etxt,')'];
  if keep_C=='Y'
    STRtitle=[STRtitle,' (GREEN component)'];
    STRxlabel=['GREEN component ',STRxlabel];
    Fused=[Fused,' green'];
  end;
  title(STRtitle);
```

```
    xlabel(STRxlabel);
    ylabel('Number of Occurrences');
    legend('Noise','Sig+Noise',0);
    if ShowDataPoints=='Y'
     Zmin=min([null_g; sig_g; 0]);
     if Zmin<0
      Tplate=SubOptNoise_g==0;
      Tplate=Tplate*Zmin;
      SubOptNoise_g=SubOptNoise_g+Tplate;
      Tplate=SubOptSignal_g==0;
      Tplate=Tplate*Zmin;
      SubOptSignal_g=SubOptSignal_g+Tplate;
      clear Tplate
     end;
     Zmax=max([null_g; sig_g; 0]);
     figure
     mesh(SubOptNoise_g)
     axis([1 on 1 om Zmin Zmax])
     axis ij
     title(['Noise for ',STRtitle]);
     ylabel('Image pixel row number');
     xlabel('Image pixel column number');
     zlabel(STRxlabel);
     figure
     mesh(SubOptSignal_g)
     axis([1 on 1 om Zmin Zmax])
     axis ij
     title(['Signal+Noise for ',STRtitle]);
     ylabel('Image pixel row number');
     xlabel('Image pixel column number');
     zlabel(STRxlabel);
    end;
  end;
  if show_b=='Y'
    maxX=abs(max([sig_b;null_b]));
    E=ceil(log10(maxX));
    Estr=int2str(E);
    Etxt=[];
    lEstr=length(Estr);
    for Epos=1:1:lEstr
     Etxt=[Etxt,'^',Estr(Epos)];
    end;
    sig_b=sig_b/(10^E);
    null_b=null_b/(10^E);
    if mean(sig_b)<mean(null_b)
     sig_b=-sig_b;
     null_b=-null_b;
     disp(' ');
     disp(' ');
     STRmsg='NOTE!: "inverse" energy situation- data inverted';
     if keep_C=='Y'
      STRmsg=[STRmsg,' for BLUE component'];
     end;
     disp(STRmsg);
```

```
 end;
 if ShowDataPoints=='Y'
  SubOptNoise_b=sparse(RowIndex,ColIndex,null_b,om,on);
  SubOptSignal_b=sparse(RowIndex,ColIndex,sig_b,om,on);
 end;
 Top=max([sig_b;null_b]);
 Bott=min([sig_b;null_b]);
 Inc=(Top-Bott)/50;
 N=Bott:Inc:Top;
 figure
 hist([null_b,sig_b],N)
 STRtitle=SourceInfo;
 STRxlabel=['2D Matched Filter Output (x10',Etxt,')'];
 if keep_C=='Y'
  STRtitle=[STRtitle,' (BLUE component)'];
  STRxlabel=['BLUE component ',STRxlabel];
  Fused=[Fused,' blue'];
 end;
 title(STRtitle);
 xlabel(STRxlabel);
 ylabel('Number of Occurrences');
 legend('Noise','Sig+Noise',0);
 if ShowDataPoints=='Y'
  Zmin=min([null_b; sig_b; 0]);
  if Zmin<0
   Tplate=SubOptNoise_b==0;
   Tplate=Tplate*Zmin;
   SubOptNoise_b=SubOptNoise_b+Tplate;
   Tplate=SubOptSignal_b==0;
   Tplate=Tplate*Zmin;
   SubOptSignal_b=SubOptSignal_b+Tplate;
   clear Tplate
  end;
  Zmax=max([null_b; sig_b; 0]);
  figure
  mesh(SubOptNoise_b)
  axis([1 on 1 om Zmin Zmax])
  axis ij
  title(['Noise for ',STRtitle]);
  ylabel('Image pixel row number');
  xlabel('Image pixel column number');
  zlabel(STRxlabel);
  figure
  mesh(SubOptSignal_b)
  axis([1 on 1 om Zmin Zmax])
  axis ij
  title(['Signal+Noise for ',STRtitle]);
  ylabel('Image pixel row number');
  xlabel('Image pixel column number');
  zlabel(STRxlabel);
 end;
end;
if show_C=='Y'
 maxX=abs(max([sig_c;null_c]));
```

```
E=ceil(log10(maxX));
Estr=int2str(E);
Etxt=[];
lEstr=length(Estr);
for Epos=1:1:lEstr
  Etxt=[Etxt,'^',Estr(Epos)];
end;
sig_c=sig_c/(10^E);
null_c=null_c/(10^E);
if mean(sig_c)<mean(null_c)
  sig_c=-sig_c;
  null_c=-null_c;
  disp(' ');
  disp(' ');
  STRmsg='NOTE!: "inverse" energy situation- data inverted';
  if keep_C=='Y'
    STRmsg=[STRmsg,' for FUSED(',Fused,')'];
  end;
  disp(STRmsg);
end;
if ShowDataPoints=='Y'
  SubOptNoise_C=sparse(RowIndex,ColIndex,null_c,om,on);
  SubOptSignal_C=sparse(RowIndex,ColIndex,sig_c,om,on);
end;
Top=max([sig_c;null_c]);
Bott=min([sig_c;null_c]);
Inc=(Top-Bott)/50;
N=Bott:Inc:Top;
figure
hist([null_c,sig_c],N)
STRtitle=SourceInfo;
STRxlabel=['3D Matched Filter Output (x10',Etxt,')'];
if keep_C=='Y'
  STRtitle=[STRtitle,' (FUSED',Fused,')'];
  STRxlabel=['FUSED (',Fused,') ',STRxlabel];
end;
title(STRtitle);
xlabel(STRxlabel);
ylabel('Number of Occurrences');
legend('Noise','Sig+Noise',0);
if ShowDataPoints=='Y'
 Zmin=min([null_c; sig_c; 0]);
 if Zmin<0
   Tplate=SubOptNoise_C==0;
   Tplate=Tplate*Zmin;
   SubOptNoise_C=SubOptNoise_C+Tplate;
   Tplate=SubOptSignal_C==0;
   Tplate=Tplate*Zmin;
   SubOptSignal_C=SubOptSignal_C+Tplate;
   clear Tplate
 end;
 Zmax=max([null_c; sig_c; 0]);
 figure
 mesh(SubOptNoise_C)
```

```
    axis([1 on 1 om Zmin Zmax])
    axis ij
    title(['Noise for ',STRtitle]);
    ylabel('Image pixel row number');
    xlabel('Image pixel column number');
    zlabel(STRxlabel);
    figure
    mesh(SubOptSignal_C)
    axis([1 on 1 om Zmin Zmax])
    axis ij
    title(['Signal+Noise for ',STRtitle]);
    ylabel('Image pixel row number');
    xlabel('Image pixel column number');
    zlabel(STRxlabel);
  end;
end;
```

M-file to calculate and display the sensitivities and bias.

Filename: **CalcSensAndBias.m**

```
%
% SUBPROGRAM:  MATLAB 5.0 m-files
% Written by:  James S. Ogawa, in support of Master's Degree Thesis work
% September, 1997, Naval Postgraduate School
% Base program supplied by: Dr. Dean Scribner, NRL
%
%    This MATLAB m-file is a sub-program of "go_MCfilter".  This code calculates
% the associated Pd and Pfa for a data field and a point evaluation and outputs
% the associated statistics and calculated sensitivity (d-prime) and response
% bias (beta).
%
% uses:
%      Threshold_at_Pt.m
%
%
disp(' ');
disp(' ');
disp('Enter the file containing the FIELD Data (".dat" assumed): ');
Fieldname=input(':: ','s');
Fieldfile=[Fieldname,'.dat'];
disp(' ');
disp(' ');
disp('Enter the file containing the POINT Data (".dat" assumed): ');
Pointname=input(':: ','s');
Pointfile=[Pointname,'.dat'];

fid=fopen(Fieldfile,'r');
Ilength=fread(fid,[1,2],'short');
Imagename=fread(fid,Ilength,'char');
Imagename=setstr(Imagename);
Tlength=fread(fid,[1,2],'short');
Targetname=fread(fid,Tlength,'char');
Targetname=setstr(Targetname);
Flength=fread(fid,[1,2],'short');
Foptions=fread(fid,Flength,'char');
Foptions=setstr(Foptions);
keep_r=setstr(fread(fid,[1,1],'char'));
keep_g=setstr(fread(fid,[1,1],'char'));
keep_b=setstr(fread(fid,[1,1],'char'));
keep_C=setstr(fread(fid,[1,1],'char'));
om=fread(fid,[1,1],'short');
on=fread(fid,[1,1],'short');
oXmin=fread(fid,[1,1],'short');
oYmin=fread(fid,[1,1],'short');
oXmax=fread(fid,[1,1],'short');
oYmax=fread(fid,[1,1],'short');
skipX=fread(fid,[1,1],'short');
skipY=fread(fid,[1,1],'short');
Sum_dim=fread(fid,[1,2],'short');
Fsummary=fread(fid,Sum_dim,'float64');
```

```
fclose(fid);
[rows,cols] = size(Fsummary);
count = rows;

fid=fopen(Pointfile,'r');
Ilength=fread(fid,[1,2],'short');
Imagename=fread(fid,Ilength,'char');
Imagename=setstr(Imagename);
Tlength=fread(fid,[1,2],'short');
Targetname=fread(fid,Tlength,'char');
Targetname=setstr(Targetname);
Flength=fread(fid,[1,2],'short');
Foptions=fread(fid,Flength,'char');
Foptions=setstr(Foptions);
keep_r=setstr(fread(fid,[1,1],'char'));
keep_g=setstr(fread(fid,[1,1],'char'));
keep_b=setstr(fread(fid,[1,1],'char'));
keep_C=setstr(fread(fid,[1,1],'char'));
om=fread(fid,[1,1],'short');
on=fread(fid,[1,1],'short');
oXmin=fread(fid,[1,1],'short');
oYmin=fread(fid,[1,1],'short');
oXmax=fread(fid,[1,1],'short');
oYmax=fread(fid,[1,1],'short');
skipX=fread(fid,[1,1],'short');
skipY=fread(fid,[1,1],'short');
Sum_dim=fread(fid,[1,2],'short');
Psummary=fread(fid,Sum_dim,'float64');
fclose(fid);

sig_r = Fsummary(:,2);
sig_g = Fsummary(:,3);
sig_b = Fsummary(:,4);
sig_c = Fsummary(:,5);

null_r = Fsummary(:,6);
null_g = Fsummary(:,7);
null_b = Fsummary(:,8);
null_c = Fsummary(:,9);

Tsig_r = Psummary(1,2);
Tsig_g = Psummary(1,3);
Tsig_b = Psummary(1,4);
Tsig_c = Psummary(1,5);

Tnull_r = Psummary(1,6);
Tnull_g = Psummary(1,7);
Tnull_b = Psummary(1,8);
Tnull_c = Psummary(1,9);

Dvector=[];
Bvector=[];

result_c=0;
```

```
 if keep_C=='Y'
  % COMPUTE SENSITIVITY, BIAS FOR (COLOR)
  disp('COMPUTING SENSITIVITY, BIAS FOR COMPOSITE (COLOR)')
  if (mean(sig_c(:)) < mean(null_c(:)))
   sig_c=-sig_c;  % corrects for "inverse" energy situations
   null_c=-null_c;
   Tsig_c=-Tsig_c;
   Tnull_c=-Tnull_c;
  end;
  [Ph_c,Pfa_c,Dprime_c, Bias_c] = Threshold_at_Pt(sig_c, null_c, count, Tsig_c, Tnull_c);
  Dvector=[Dvector; Dprime_c];
  Bvector=[Bvector; Bias_c];
 end;

 result_r=0;
 if keep_r=='Y'
  % COMPUTE SENSITIVITY, BIAS FOR (RED)
  disp('COMPUTING SENSITIVITY, BIAS FOR (RED)')
  if (mean(sig_r(:)) < mean(null_r(:)))
   sig_r=-sig_r;  % corrects for "inverse" energy situations
   null_r=-null_r;
   Tsig_r=-Tsig_r;
   Tnull_r=-Tnull_r;
  end;
  [Ph_r,Pfa_r,Dprime_r, Bias_r] = Threshold_at_Pt(sig_r, null_r, count, Tsig_r, Tnull_r);
  Dvector=[Dvector; Dprime_r];
  Bvector=[Bvector; Bias_r];
 end;

 result_g=0;
 if keep_g=='Y'
  % COMPUTE SENSITIVITY, BIAS FOR (GREEN)
  disp('COMPUTING SENSITIVITY, BIAS (GREEN)')
  if (mean(sig_g(:)) < mean(null_g(:)))
   sig_g=-sig_g;  % corrects for "inverse" energy situations
   null_g=-null_g;
   Tsig_g=-Tsig_g;
   Tnull_g=-Tnull_g;
  end;
  [Ph_g,Pfa_g,Dprime_g, Bias_g] = Threshold_at_Pt(sig_g, null_g, count, Tsig_g, Tnull_g);
  Dvector=[Dvector; Dprime_g];
  Bvector=[Bvector; Bias_g];
 end;

 result_b=0;
 if keep_b=='Y'
  % COMPUTE SENSITIVITY, BIAS FOR (BLUE)
  disp('COMPUTING SENSITIVITY, BIAS FOR (BLUE)')
  if (mean(sig_b(:)) < mean(null_b(:)))
   sig_b=-sig_b;  % corrects for "inverse" energy situations
   null_b=-null_b;
   Tsig_b=-Tsig_b;
   Tnull_b=-Tnull_b;
  end;
```

```
  [Ph_b,Pfa_b,Dprime_b, Bias_b] = Threshold_at_Pt(sig_b, null_b, count, Tsig_b, Tnull_b);
  Dvector=[Dvector; Dprime_b];
  Bvector=[Bvector; Bias_b];
end;

Dvector
Bvector
```

APPENDIX K. S-PLUS FUNCTION TO PRODUCE SENSITIVITIES

Filename: **CalcSensAndBias (·)**

```
function(TestData)
{
#    This function takes the formatted data results of
# the human factors testing and computes each
# individual's sensitivity (d') and bias (Beta) as well
# as the overall sensitivity and bias per image class.
# The data must be arranged into six columns in the
# following order: [V1] Names, [V2] Sensor, [V3] Scene,
# [V4] Position, [V5] Condition Shown, [V6] Subject's
# response.
#    This code evaluates the sensitivities by the number
# of positions which the target was placed at.  Thus, in
# the case of three target positions, those scenes without
# a target present will have to be matched to respective
# target positions.  Thus if the target was placed at
# position "1", "2", or "3", then the corresponding scenes
# without a target could be labelled "n1", "n2", or "n3".
# In any case, the code assumes that the first half of the
# alphanumerically ordered "levels" in [V4] corresponds to
# the various positions of the target, and the lower half
# corresponding to the same scene shown without a target.
#    The conditions shown were "A" for a target present
# in the image, and "B" for no target present in the scene.
#    This code automatically adjusts for any number of
# subjects, any number of scenes, as well as any number of
# sensors.
attach(TestData)
trials <- length(V1)
classes <- c(levels(V2))
images <- c(levels(V3))
position <- c(levels(V4))
Xdim <- (length(images) * length(classes))
Xnames <- rep("*", Xdim)
Iindex <- 0
Cindex <- length(classes)
repeat {
 Range1 <- (Iindex * length(classes)) + 1
 Range2 <- (Range1 + Cindex - 1)
 Xnames[Range1:Range2] <- paste(images[Iindex + 1], classes)
 if(Iindex == length(images) - 1)
  break
 Iindex <- Iindex + 1
}
Names <- levels(V1)
Ynames <- c(Names, "overall")
Ydim <- length(Ynames)
Zdim <- length(position)
Zdim2 <- Zdim/2
Znames <- position
```

```
 Znames2 <- paste(position[1:Zdim2], position[(Zdim2 + 1):Zdim])
 ARinit1 <- array(0, dim = c(Xdim, Ydim, Zdim), dimnames = list(Xnames,
  Ynames, Znames))
 ARinit2 <- array(0, dim = c(Xdim, Ydim, Zdim/2), dimnames = list(
  Xnames, Ynames, Znames2))
 AA <- ARinit1
 AB <- ARinit1
 BB <- ARinit1
 BA <- ARinit1
 Ph <- ARinit2
 Pfa <- ARinit2
 QnormPh <- ARinit2
 QnormPfa <- ARinit2
 Dprime <- ARinit2
 DnormPh <- ARinit2
 DnormPFa <- ARinit2
 Bias <- ARinit2
 for(Pos in position) {
  cat("working segment ", Pos, " of {", paste(position), "}","\n")
  for(Im in images)
   for(Clss in classes)
    for(Nm in Names) {
     ImClss <- paste(Im, Clss)
     Select <- (V2 == Clss) * (V3 == Im) * (V4 == Pos) * (V1 == Nm)
AA[ImClss, Nm, Pos] <- sum(Select * (V5 == "A") * (V6 == "A"))
     AA[ImClss, "overall", Pos] <- AA[ImClss, "overall", Pos] + AA[ImClss, Nm, Pos]
     AB[ImClss, Nm, Pos] <- sum(Select * (V5 == "A") * (V6 == "B"))
     AB[ImClss, "overall", Pos] <- AB[ImClss, "overall", Pos] + AB[ImClss, Nm, Pos]
     BB[ImClss, Nm, Pos] <- sum(Select * (V5 == "B") * (V6 == "B"))
     BB[ImClss, "overall", Pos] <- BB[ImClss, "overall", Pos] + BB[ImClss, Nm, Pos]
     BA[ImClss, Nm, Pos] <- sum(Select * (V5 == "B") * (V6 == "A"))
     BA[ImClss, "overall", Pos] <- BA[ImClss, "overall", Pos] + BA[ImClss, Nm, Pos]
            }
           }
  detach(2)
  ALLtotals <- AA + AB + BB + BA
  Element <- ALLtotals[1, 1, 1]
  Ph <- (1/Element) * AA[, , (1:Zdim2)]
  Ph[, "overall", ] <- Ph[, "overall", ]/(Ydim - 1)
  Pfa <- (1/Element) * BA[, , ((Zdim2 + 1):Zdim)]
  Pfa[, "overall", ] <- Pfa[, "overall", ]/(Ydim - 1)
  AdjOnes <- (Ph == 1) * (0.01)
  Ph <- Ph - AdjOnes
  AdjZeros <- (Ph == 0) * (0.01)
  Ph <- Ph + AdjZeros
  AdjOnes <- (Pfa == 1) * (0.01)
  Pfa <- Pfa - AdjOnes
  AdjZeros <- (Pfa == 0) * (0.01)
  Pfa <- Pfa + AdjZeros
  QnormPh <- array(qnorm(Ph), dim = c(Xdim, Ydim, Zdim/2), dimnames= list(Xnames, Ynames,
Znames2))
  QnormPfa <- array(qnorm(Pfa), dim = c(Xdim, Ydim, Zdim/2), dimnames = list(Xnames, Ynames,
Znames2))
  Dprime <- QnormPh - QnormPfa
```

```
 DnormPh <- array(dnorm(Ph), dim = c(Xdim, Ydim, Zdim/2), dimnames = list(Xnames, Ynames,
Znames2))
 DnormPfa <- array(dnorm(Pfa), dim = c(Xdim, Ydim, Zdim/2), dimnames = list(Xnames, Ynames,
Znames2))
 Bias <- (DnormPh/DnormPfa)
 list(BB = BB, BA = BA, AA = AA, AB = AB, Dprime = Dprime, Bias = Bias, Ph = Ph, Pfa = Pfa,
QnormPh = QnormPh, QnormPfa = QnormPfa, DnormPh = DnormPh, DnormPfa = DnormPfa)
}
```

APPENDIX L. C++ PROGRAM FOR SPECTRAL REWEIGHTING

Filename: **FFT.C**

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include "imgio.c"
#include "four1.c"
#include "miscfops.c"

#define MAX_XSIZE 1024
#define MAX_YSIZE 1024

//WARNING: ALWAYS keep MAX_XSIZE bigger than or equal to MAX_YSIZE

//Real dimensions of image
int xsize;
int ysize;

//original and transformed images
float orig_image[MAX_YSIZE][MAX_XSIZE];
float real_image[MAX_YSIZE][MAX_XSIZE];
float imag_image[MAX_YSIZE][MAX_XSIZE];
float mag_image[MAX_YSIZE][MAX_XSIZE];
float phase_image[MAX_YSIZE][MAX_XSIZE];

float xarray[2*MAX_XSIZE];
float yarray[2*MAX_YSIZE];

float magnitude[MAX_XSIZE];
float logmag[MAX_XSIZE];
float logfreq[MAX_XSIZE];
int   freqcount[MAX_XSIZE];

void main(int argc, char **argv) {

 int lw_xsize, lw_ysize;

 //variables in for loops
 int j;

 //computed parameters of input image
 float a,b;          //linear regression coefficients
 float luminance;
 float contrast;
 float power;
 int cutoff;
 int thresh;
 float noise;

 //Input and output files
 FILE *f, *g;
```

```
int err=0;

int invplot = 0;  // when invplot=1, the inverse fft is computed.

if (!(f=fopen(argv[1],"rb"))) {
  fprintf(stderr,"Could not open input file\n");
  exit(1);
}

if (!(g=fopen(argv[2],"wb"))) {
  fprintf(stderr,"Could not open output file\n");
  exit(1);
}

read_and_normalize_pbm_image(f, orig_image, &lw_xsize, &lw_ysize, &err);

xsize=lw_xsize; ysize=lw_ysize;

copy_image(orig_image,real_image,xsize,ysize);

spat_stat_image(real_image, &luminance, &contrast, xsize, ysize);    //spatial statistics

clear_image(imag_image,xsize,ysize);

rearrange_image(real_image,xsize,ysize);

//FFT along x-direction
fprintf(stderr,"\nFFT X:\n");
for (j=0;j<ysize;j++) {
        in_array(real_image, imag_image, xarray, xsize, j, 'x');
  fft(xarray-1,xsize,1);
        out_array(real_image,imag_image, xarray, xsize, j, 'x');
}

scale_image(real_image,(float)xsize, xsize,ysize);
scale_image(imag_image,(float)xsize, xsize,ysize);
//END

//FFT along y-direction
fprintf(stderr,"\n\nFFT Y:\n");
for (j=0;j<xsize;j++) {
        in_array(real_image, imag_image, yarray, ysize, j, 'y');
  fft(yarray-1,ysize,1);
        out_array(real_image,imag_image, yarray, ysize, j, 'y');
}

scale_image(real_image,(float)ysize, xsize,ysize);
scale_image(imag_image,(float)ysize, xsize,ysize);

//convert to polar coordinates
clear_image(mag_image, xsize, ysize);
clear_image(phase_image, xsize, ysize);
polar_image(real_image, imag_image, mag_image, phase_image, xsize, ysize);
```

```
  //Compute frequency based image statistics here
  rad_freq_image(mag_image, magnitude, freqcount, &cutoff, xsize, ysize);          //compute image
magnitudes as function of distance
  //thresh=cutoff;
  fprintf(stderr,"Enter threshold:");
  scanf("%d",&thresh);
  lin_reg_image(magnitude, freqcount, thresh, cutoff, &a, &b, logfreq, logmag, &power, &noise);
//Linear regression cutoff=100 (second argument passed)

  //Write everything to a file
  fprintf(stderr,"a=%f b=%f lum=%f contr=%f pow=%f noise=%f\n",
a,b,luminance,contrast,power,10e6*noise);
  write2file(luminance, contrast, power, noise, a, b, logfreq, logmag, cutoff);

  if (invplot==1) {
   //convert back to cartesian coordinates
   clear_image(real_image, xsize, ysize);
   clear_image(imag_image, xsize, ysize);
   cartesian_image(mag_image, phase_image, real_image, imag_image, xsize, ysize);

  //************USED ONLY FOR INVERSE FFT **********

   //Inverse FFT along x-direction
   fprintf(stderr,"\nInvFFT X:\n");
   for (j=0;j<ysize;j++) {
          in_array(real_image, imag_image, xarray, xsize, j, 'x');
    fft(xarray-1,xsize,-1);
          out_array(real_image,imag_image, xarray, xsize, j, 'x');
   }
   //END

   //Inverse FFT along y-direction
   fprintf(stderr,"\nInvFFT Y:\n");
   for (j=0;j<xsize;j++) {
          in_array(real_image, imag_image, yarray, ysize, j, 'y');
    fft(yarray-1,ysize,-1);
          out_array(real_image,imag_image, yarray, ysize, j, 'y');
   }
   //END

   rearrange_image(real_image,xsize,ysize);

   limit_pixvals_image(real_image,xsize,ysize);

   write_and_denormalize_pgm_image(g, real_image, xsize, ysize, 0);  //change to pnm after debugging

   fclose(g);
  }        //end of if invplot==1

  fclose(f);
  exit(0);

}
```

APPENDIX M. C++ PROGRAM TO SPLIT IMAGES INTO RGB IMAGES

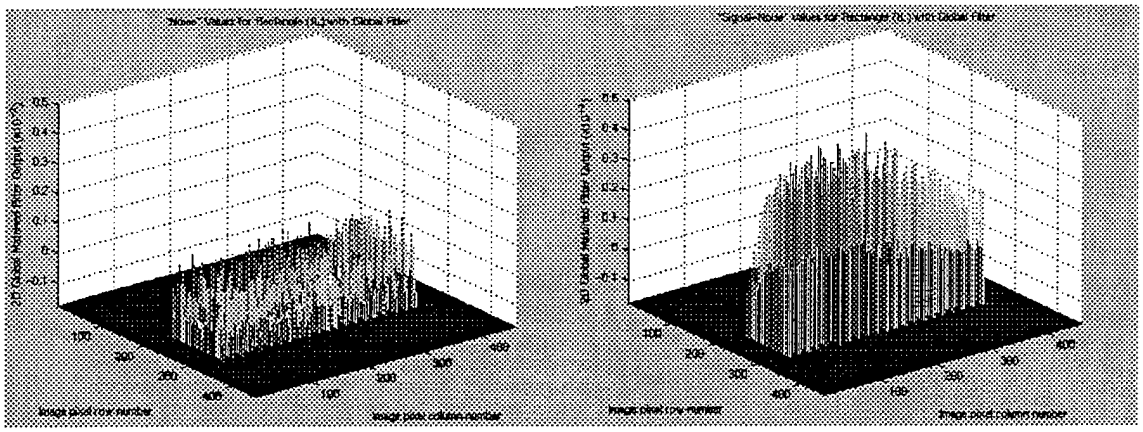
Filename: **SPLITRGB.C**

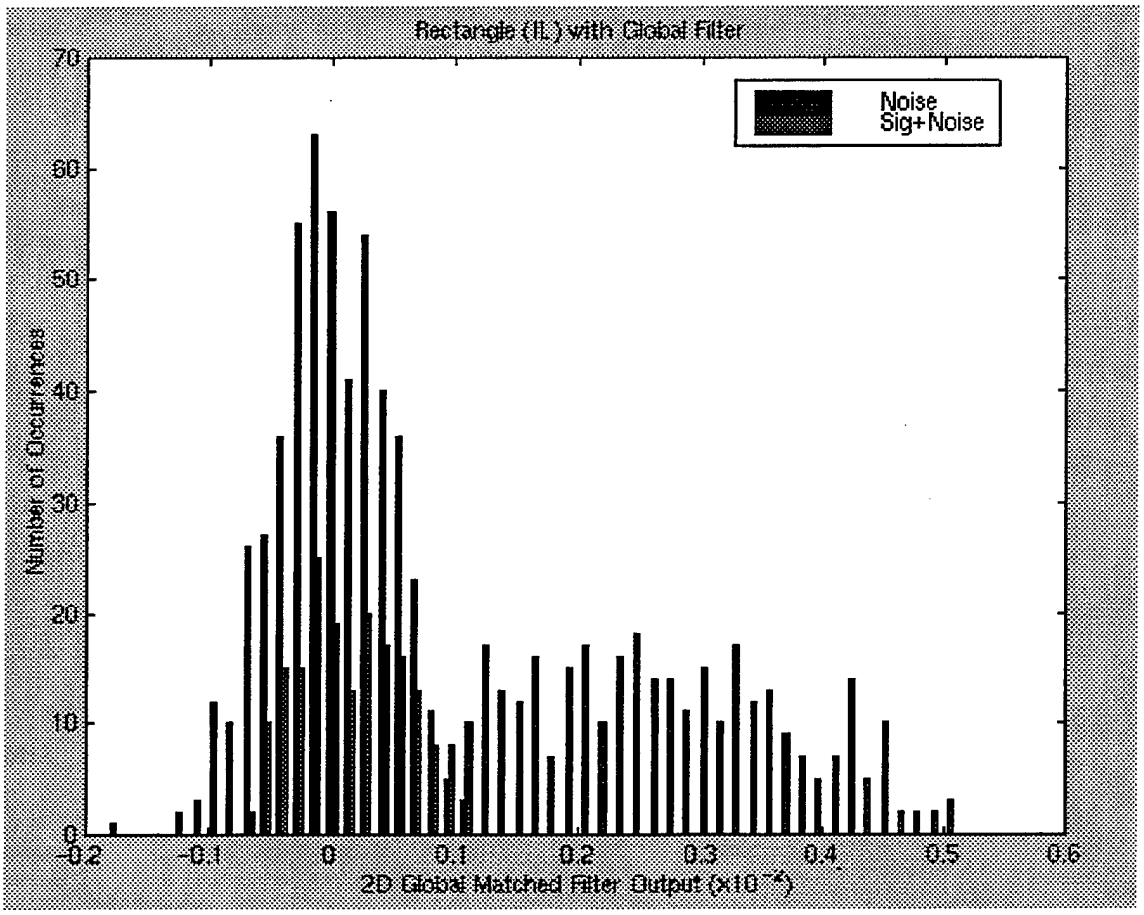
```
#include <math.h>
#include <stdio.h>
#include "imgio.c"

#define MAX_XSIZE 1024
#define MAX_YSIZE 1024

//Real dimensions of image
int xsize, ysize;
float min_time,max_time,inc_time;   //Not required in this program

//original and transformed images
float image_red[MAX_YSIZE][MAX_XSIZE];
float image_green[MAX_YSIZE][MAX_XSIZE];
float image_blue[MAX_YSIZE][MAX_XSIZE];

void main(int argc, char **argv) {

 //Input and output files
 FILE *f, *r, *g, *b;

 int err=0;

 if (!(f=fopen(argv[1],"rb"))) {
  fprintf(stderr,"Could not open input file\n");
  exit(1);
 }

 if (!(r=fopen(argv[2],"wb"))) {
  fprintf(stderr,"Could not open output RED file\n");
  exit(1);
 }
 if (!(g=fopen(argv[3],"wb"))) {
  fprintf(stderr,"Could not open output BLUE file\n");
  exit(1);
 }
 if (!(b=fopen(argv[4],"wb"))) {
  fprintf(stderr,"Could not open output GREEN file\n");
  exit(1);
 }

 read_and_normalize_ppm_image(f, image_blue, image_red, image_green, &xsize, &ysize, &err);
//BUG!!! COME DEBUG LATER!
 fprintf(stderr,"Dimensions of image: Width(X)=%d Height(Y)=%d\n",xsize,ysize);

 write_and_denormalize_pnm_image(r, image_red, xsize, ysize, 0);
 write_and_denormalize_pnm_image(g, image_green, xsize, ysize, 0);
 write_and_denormalize_pnm_image(b, image_blue, xsize, ysize, 0);
```

```
  fclose(f);
  fclose(r);
  fclose(g);
  fclose(b);
  exit(0);
}
```

APPENDIX N. COLLECTED DATA

"Rectangle" (IL) with Global matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

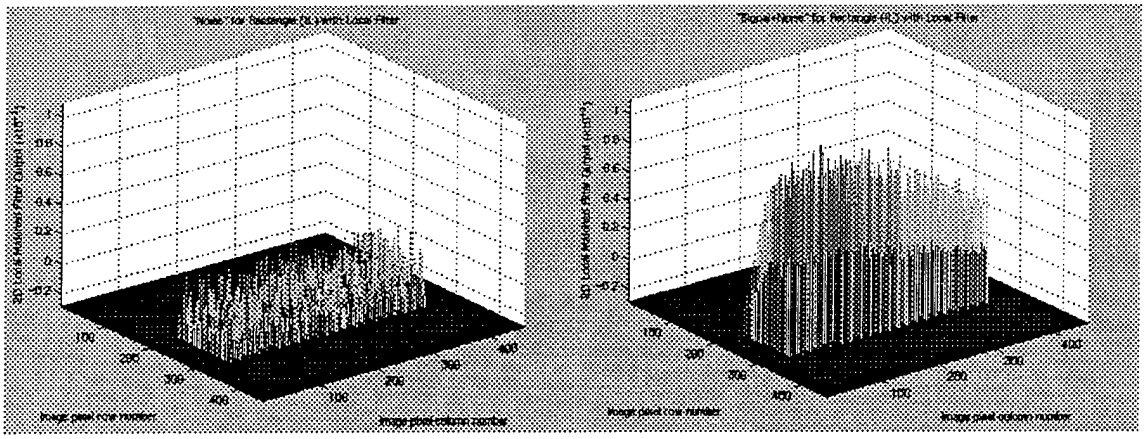


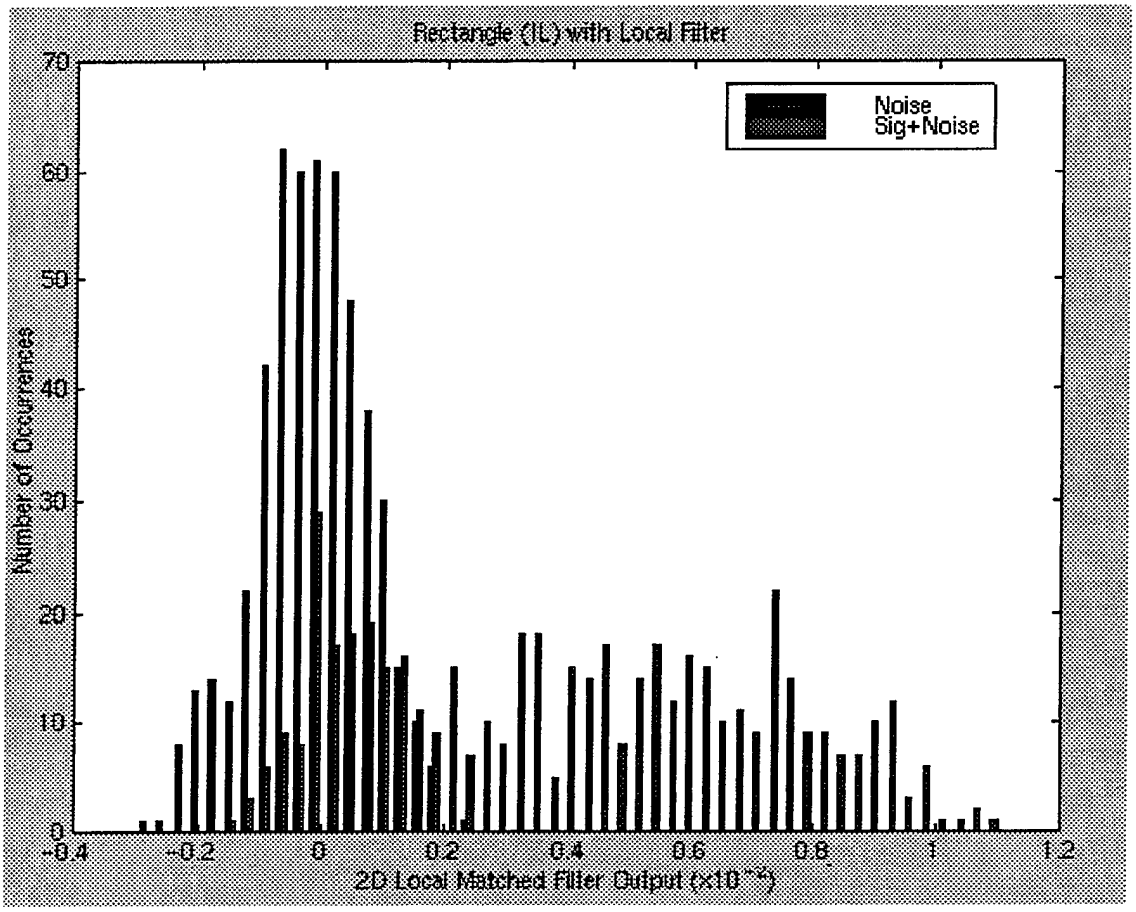


Threshold Criteria

Target Position	SNt	Nt
1	-9.22e-06	0.000814
2	-0.000296	0.000538
3	-6.43e-05	0.000604

"Rectangle" (IL) with Local matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

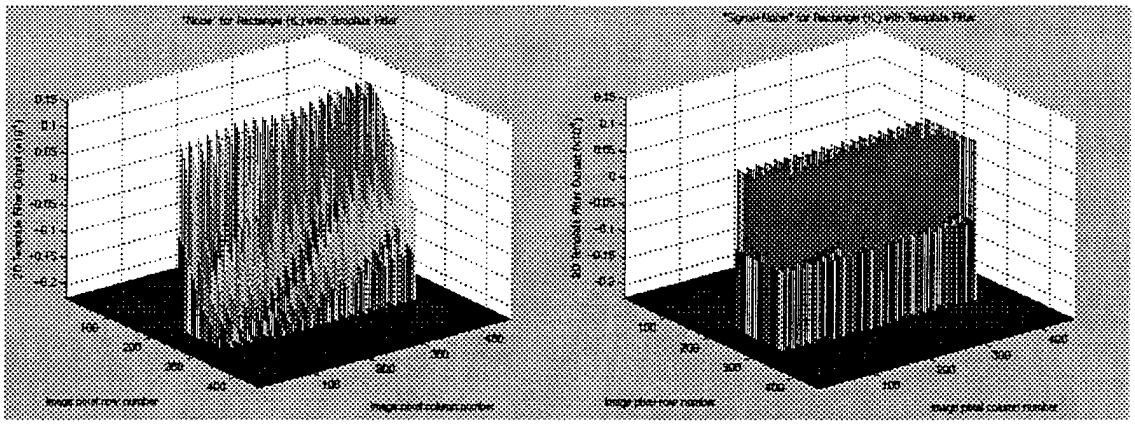


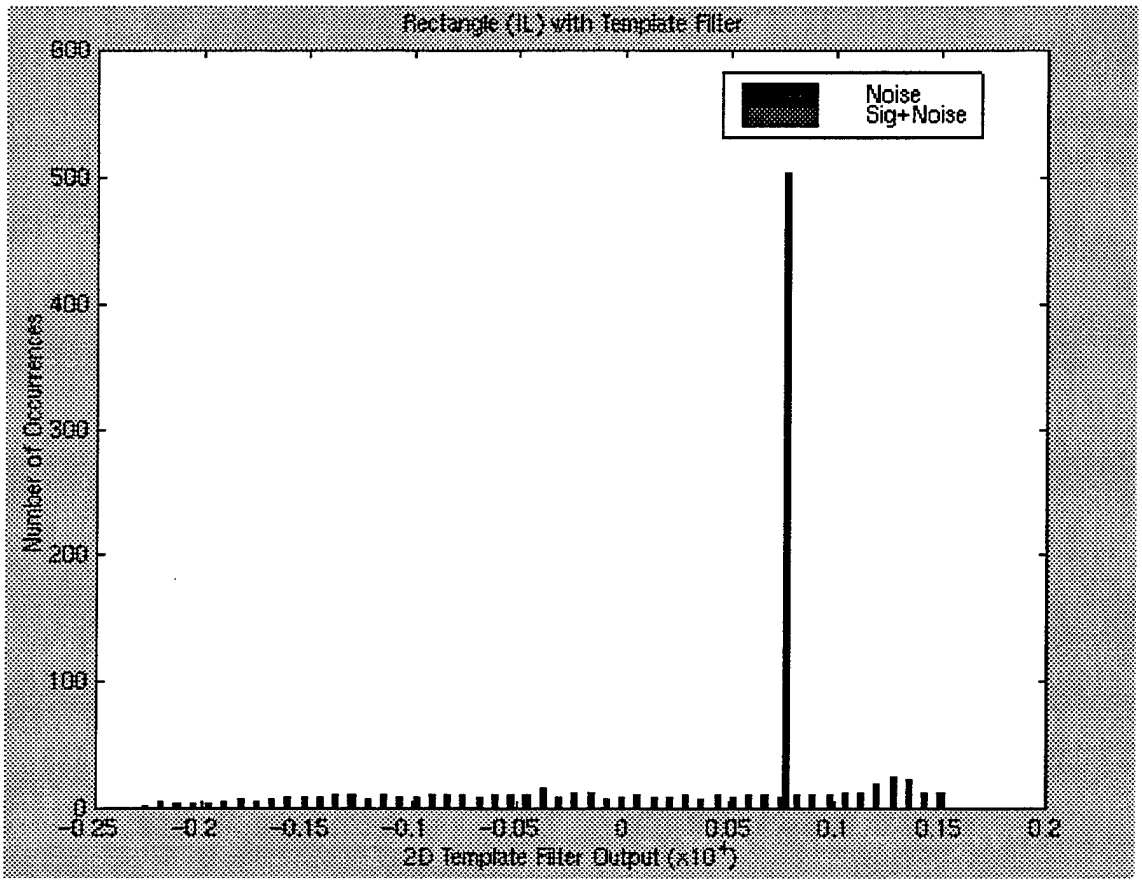


Threshold Criteria

Target Position	SNt	Nt
1	0.000643	0.00122
2	-8.45e-05	0.000872
3	0.000182	0.000976

"Rectangle" (IL) with Template matching filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

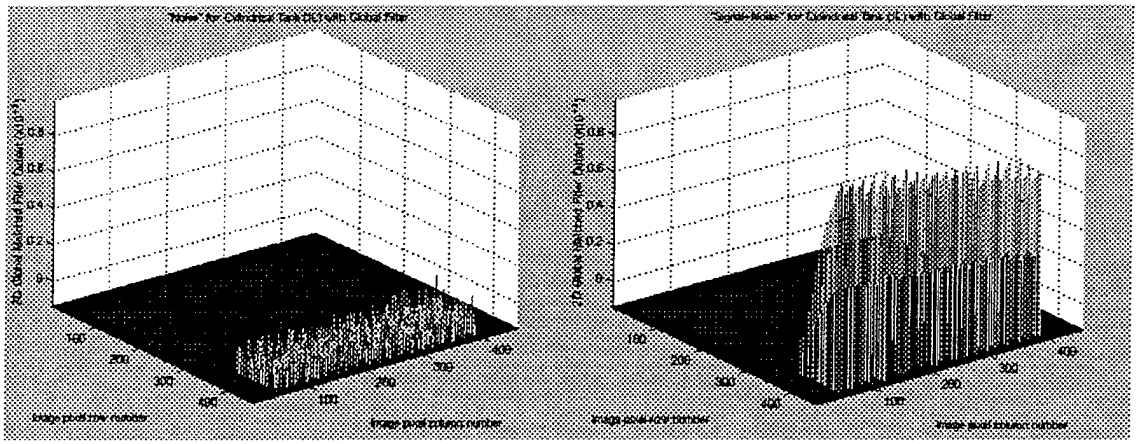


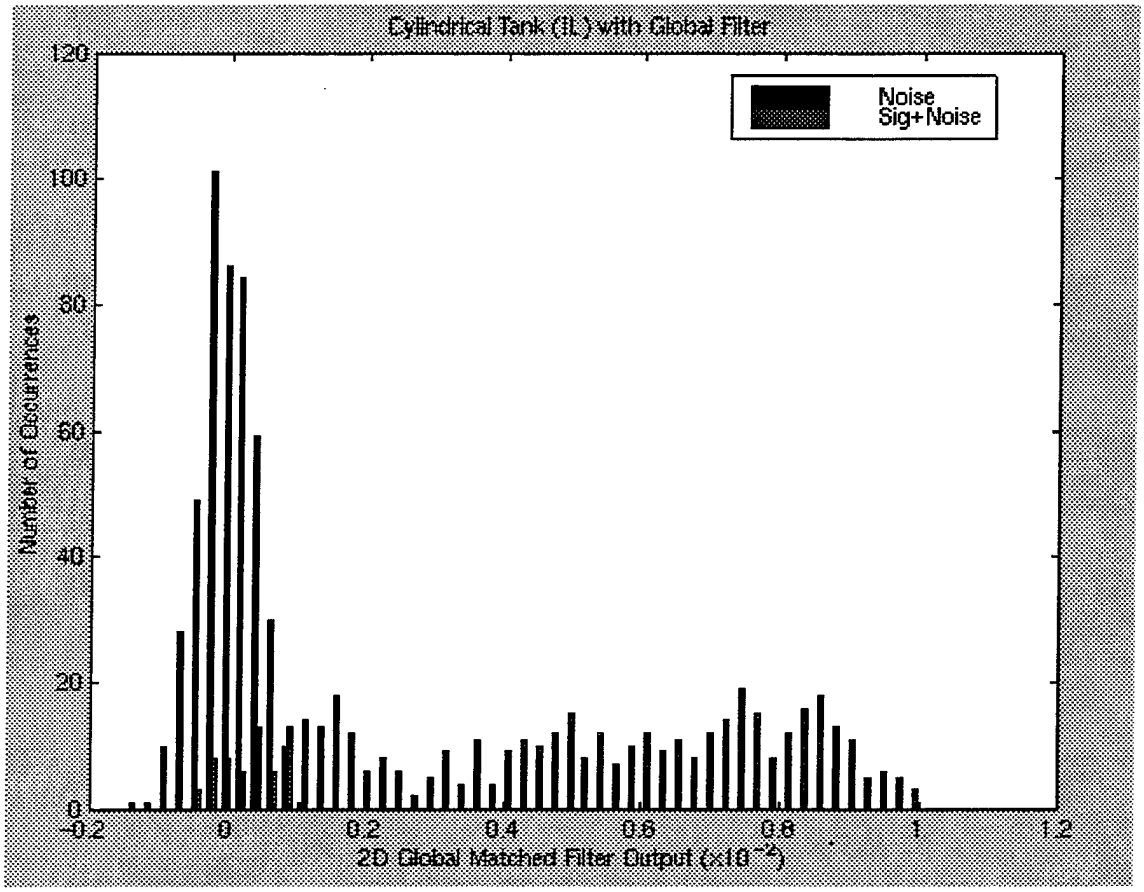


Threshold Criteria

Target Position	SNt	Nt
1	769	1.26e+03
2	769	1.31e+03
3	769	1.28e+03

"Cylindrical Tank" (IL) with Global matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

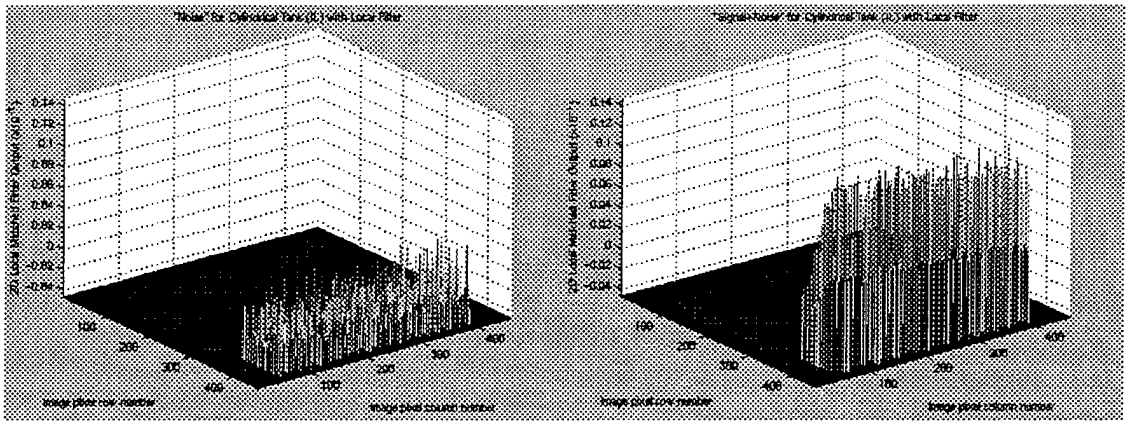


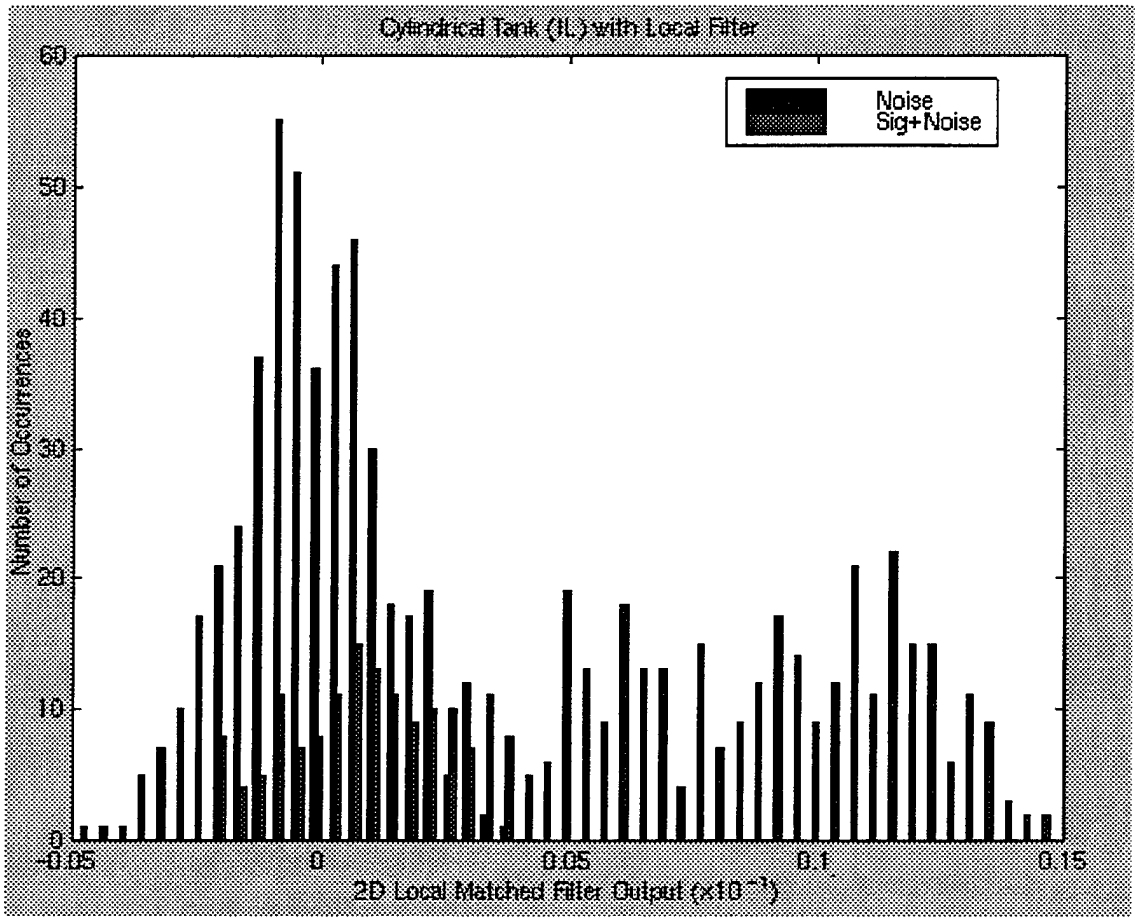


Threshold Criteria

Target Position	SNt	Nt
1	0.000813	-0.000204
2	0.000732	0.000359
3	-0.000104	-0.000181

"Cylindrical Tank" (IL) with Local matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

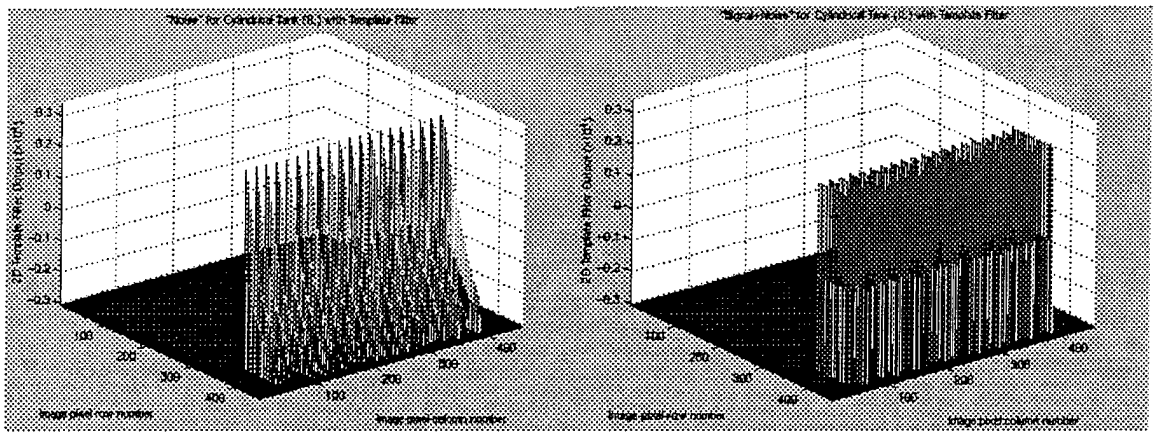


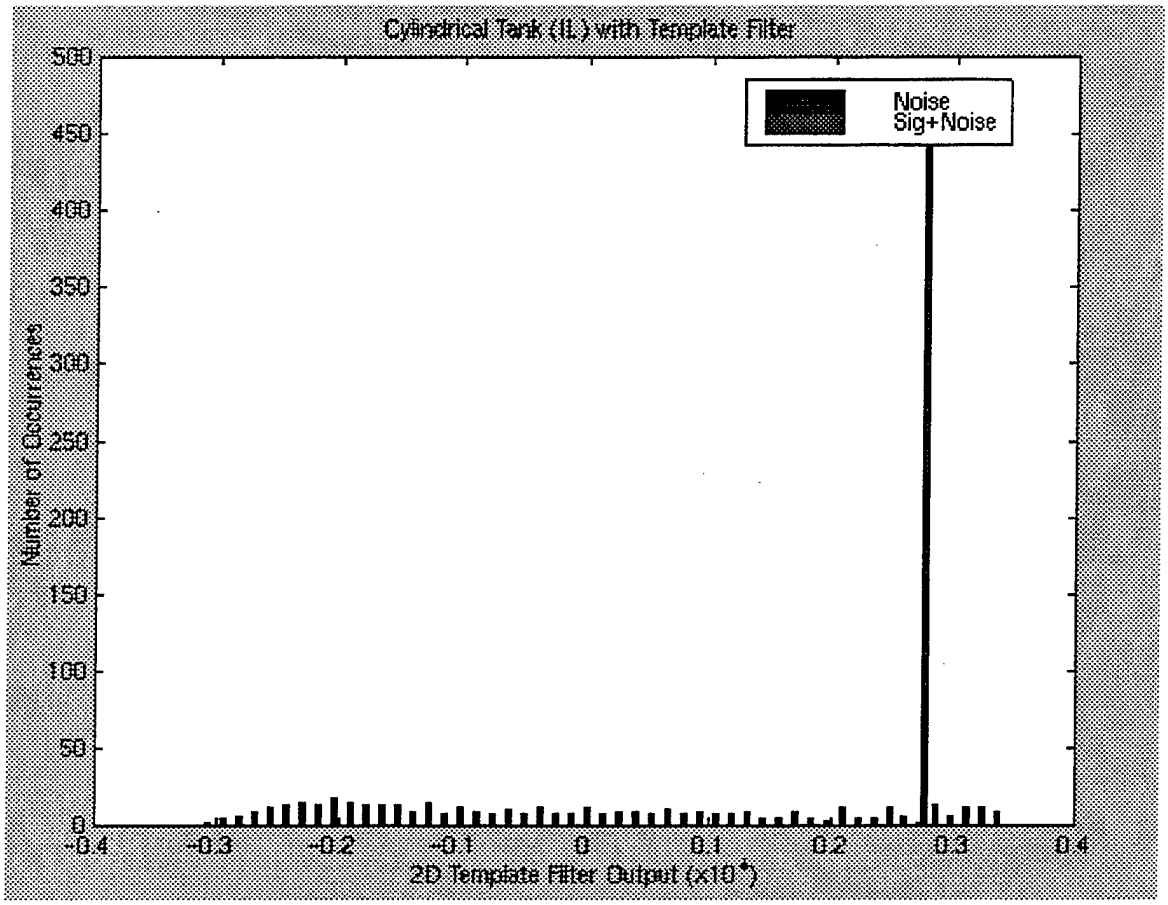


Threshold Criteria

Target Position	SNt	Nt
1	-0.00175	-0.00378
2	-0.00185	-0.00122
3	-0.00219	-0.0023

"Cylindrical Tank" (IL) with Template matching filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

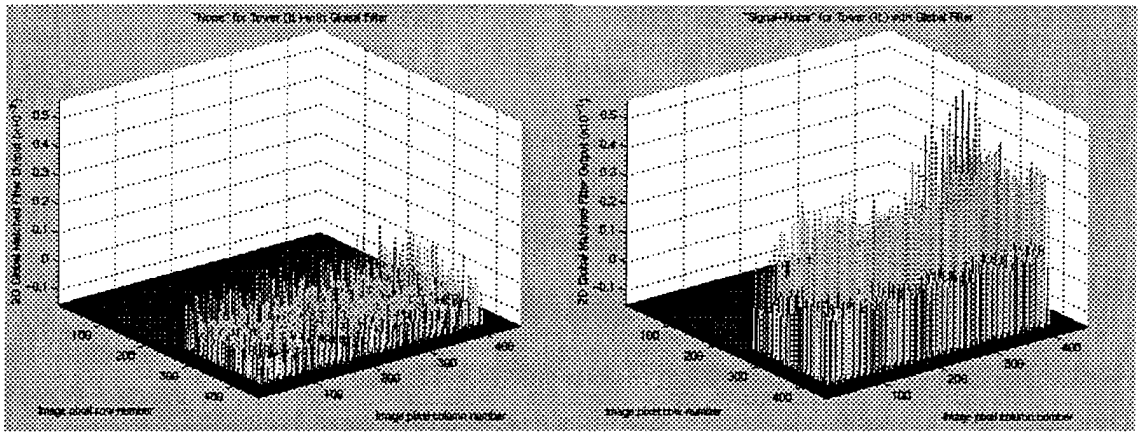


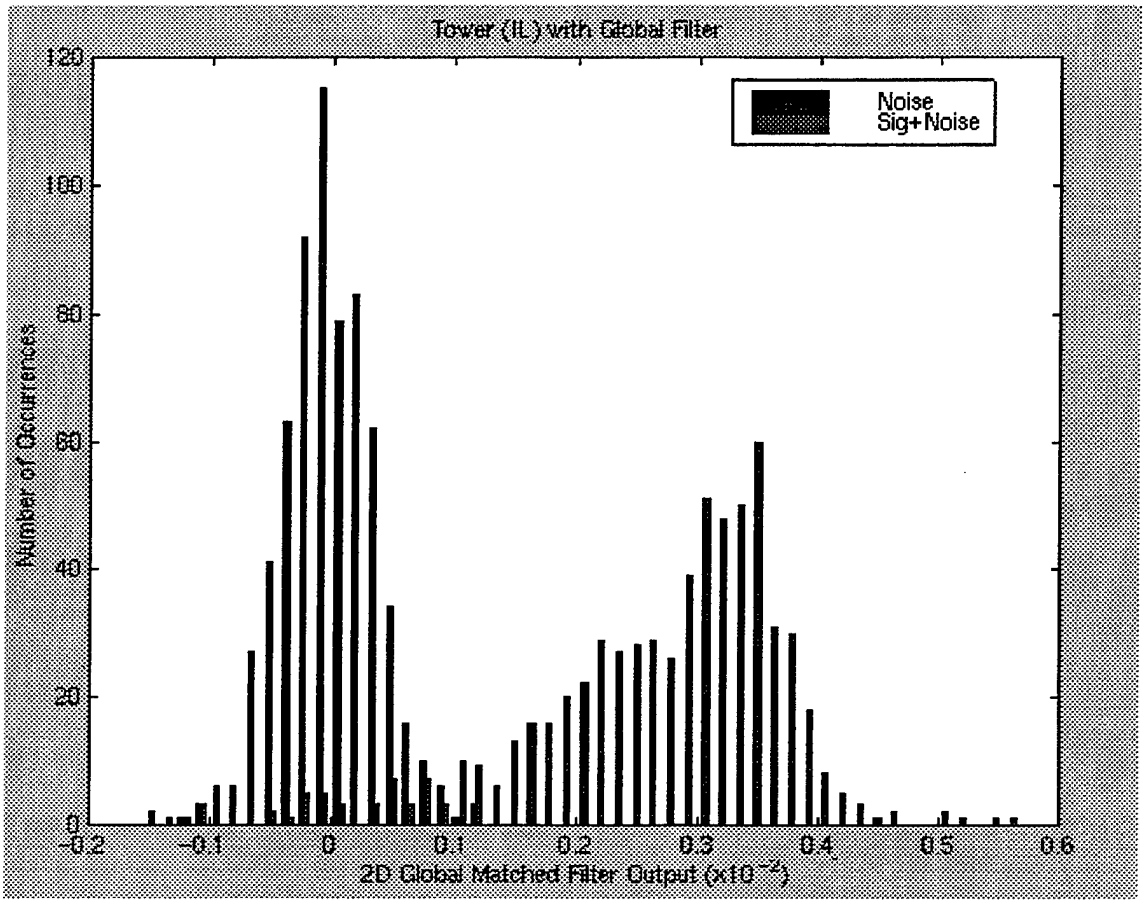


Threshold Criteria

Target Position	SNt	Nt
1	2.81e+03	2.89e+03
2	2.81e+03	3.1e+03
3	2.81e+03	3.26e+03

"Tower" (IL) with Global matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

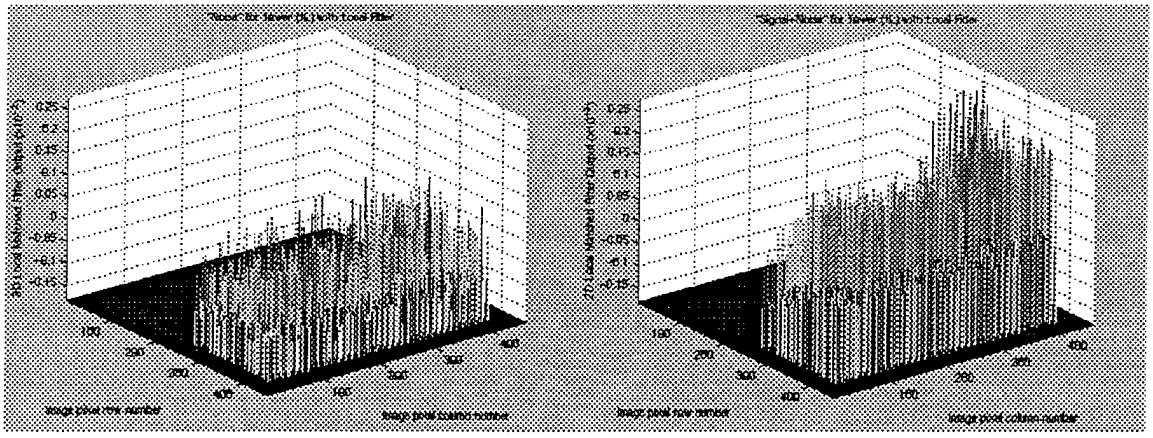


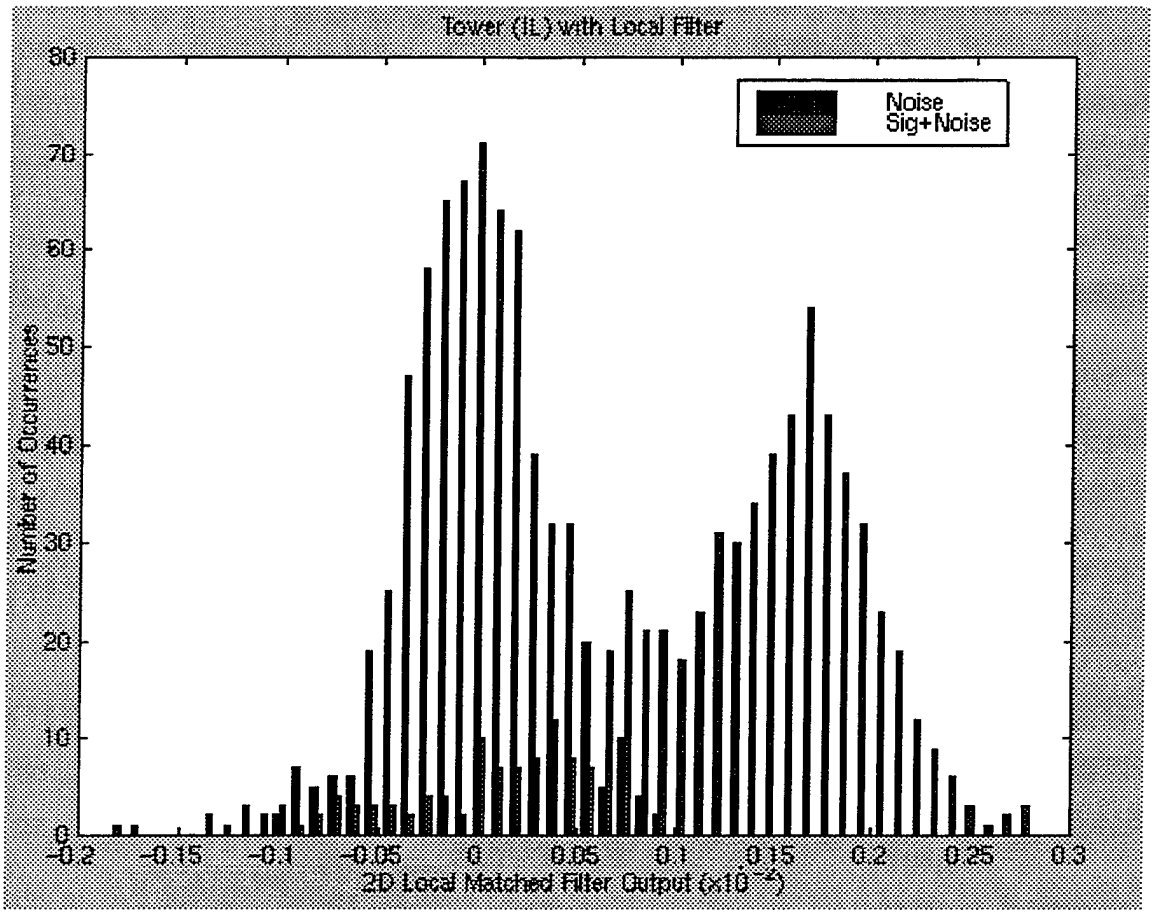


Threshold Criteria

Target Position	SNt	Nt
1	0.00259	0.000318
2	0.00331	0.000249
3	0.00284	0.000509

"Tower" (IL) with Local matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

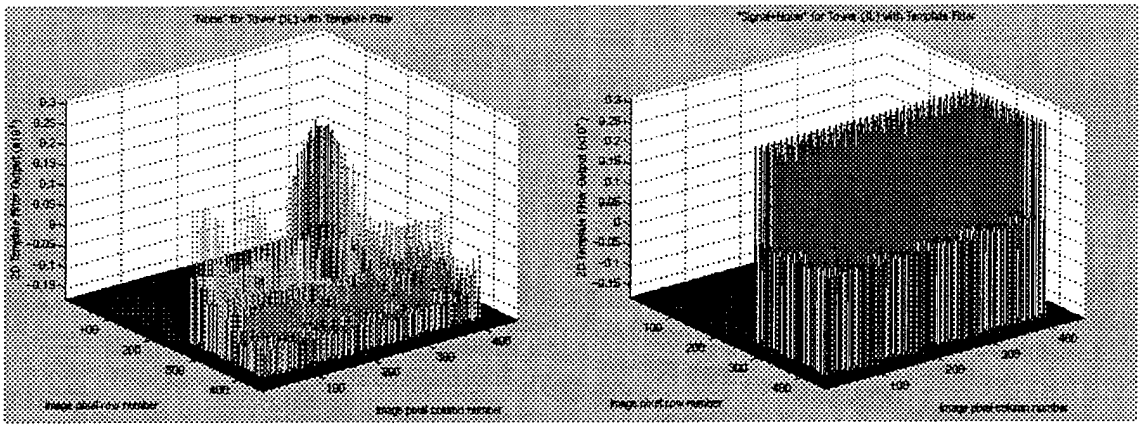


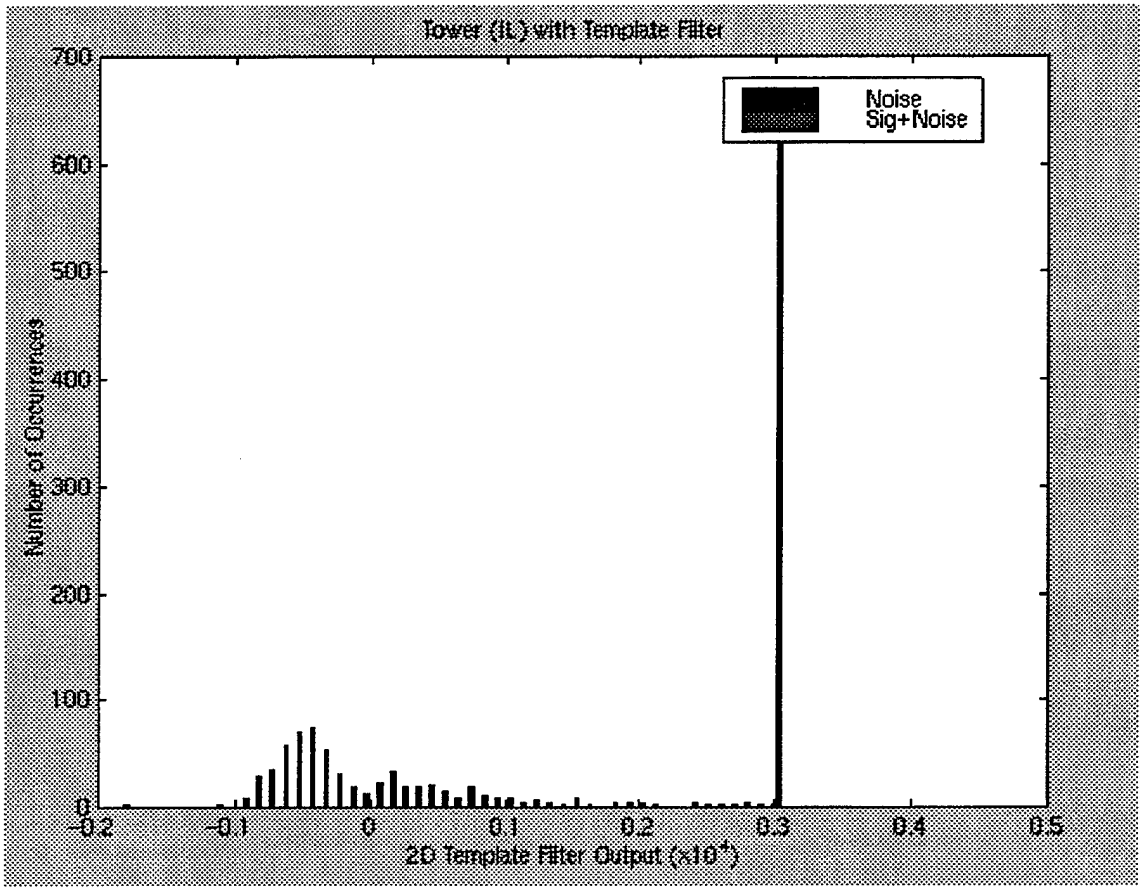


Threshold Criteria

Target Position	SNt	Nt
1	0.00178	0.000735
2	0.00165	5.81e-05
3	0.00142	-8.92e-05

"Tower" (IL) with Template matching filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

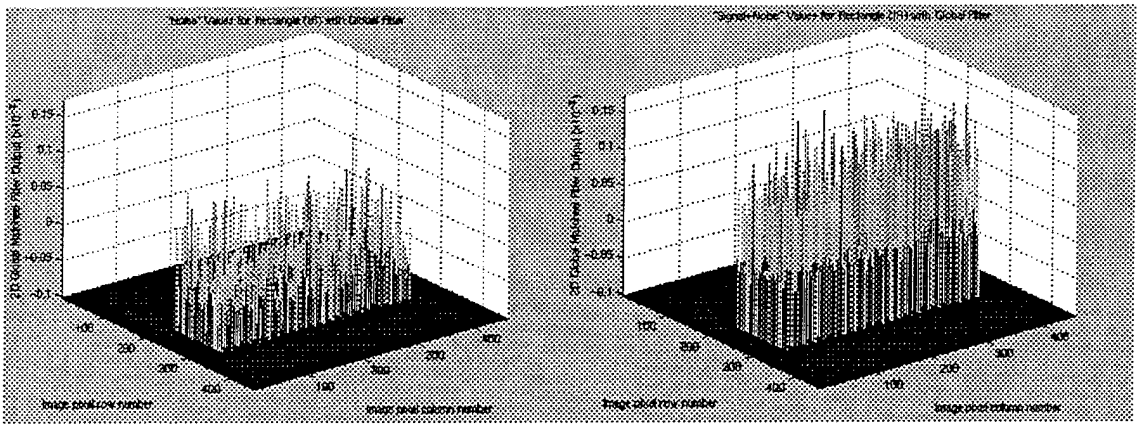


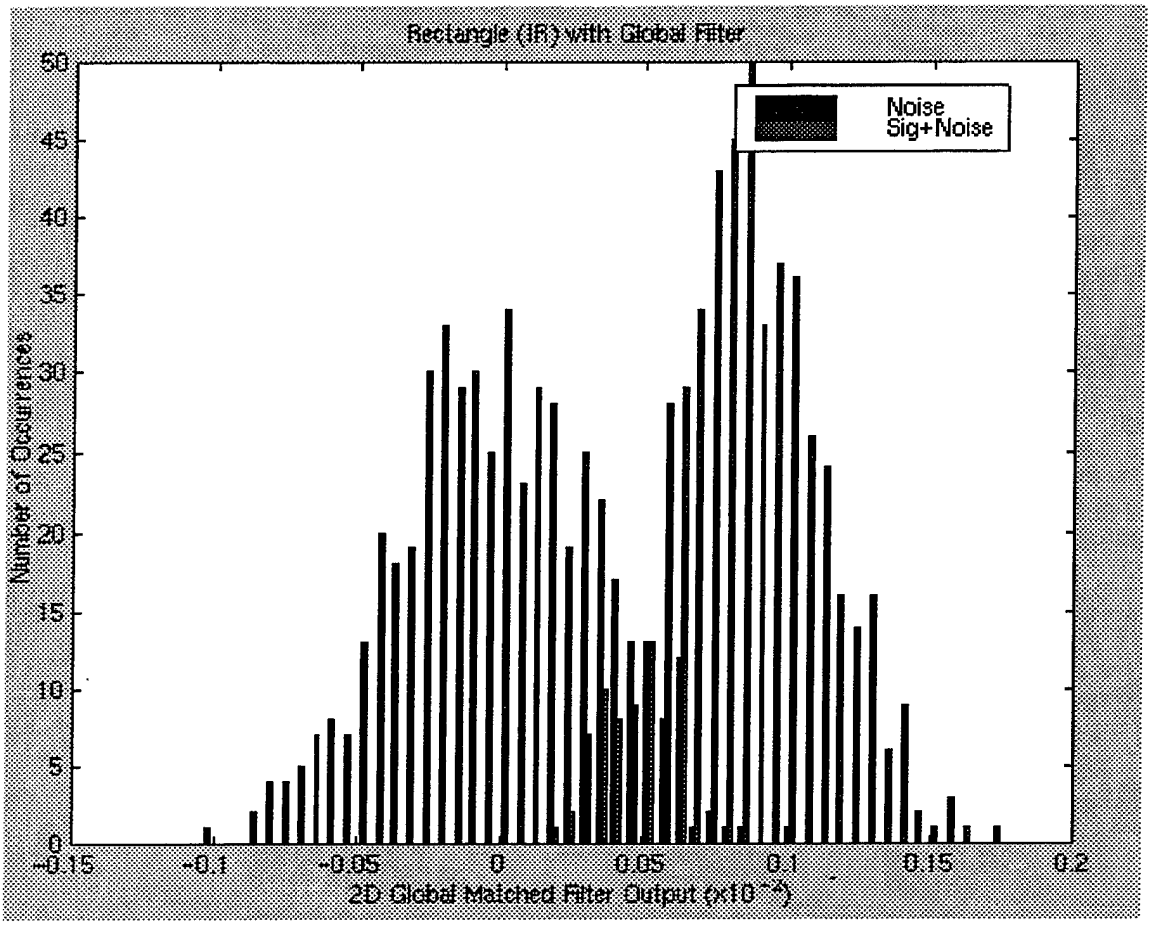


Threshold Criteria

Target Position	SNt	Nt
1	3e+03	92.2
2	3e+03	-523
3	3e+03	-297

"Rectangle" (IR) with Global matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

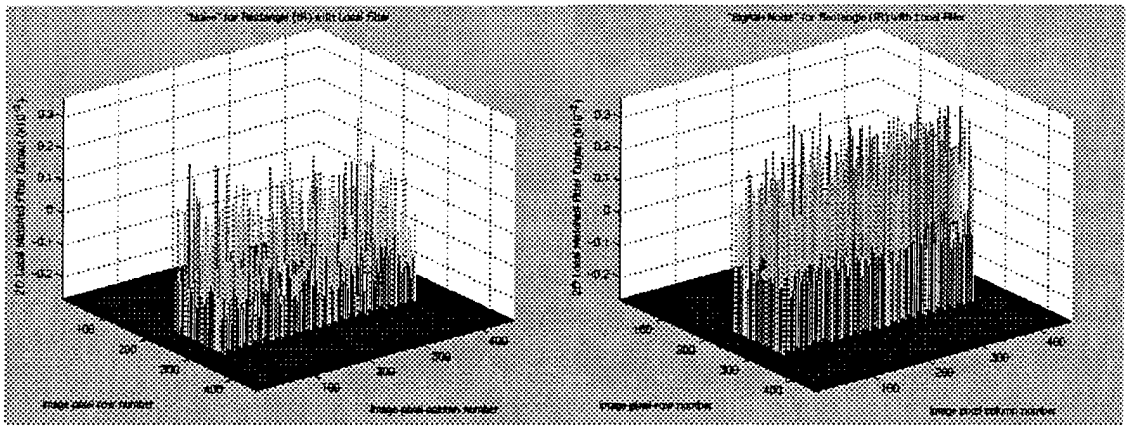


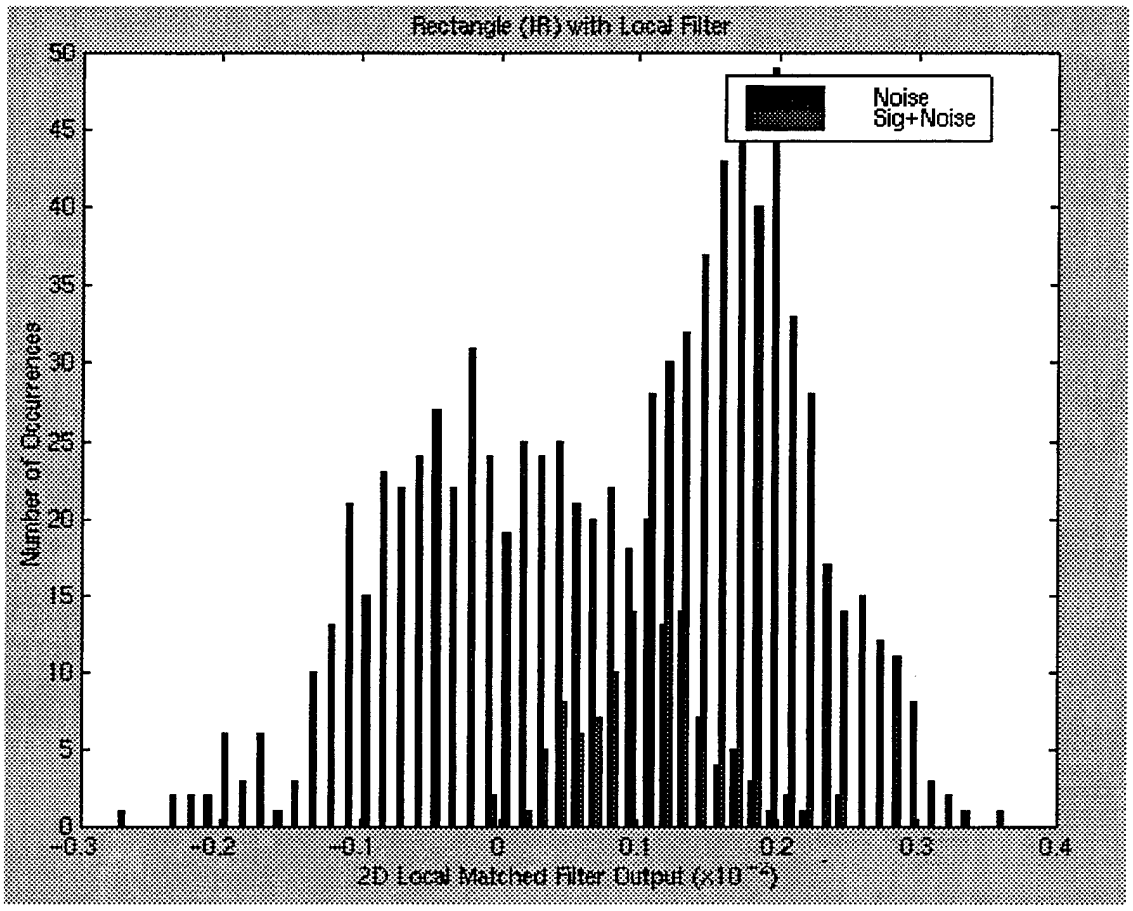


Threshold Criteria

Target Position	SNt	Nt
1	0.000702	0.000137
2	0.00101	-0.000644
3	0.00123	2.49e-05

"Rectangle" (IR) with Local matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

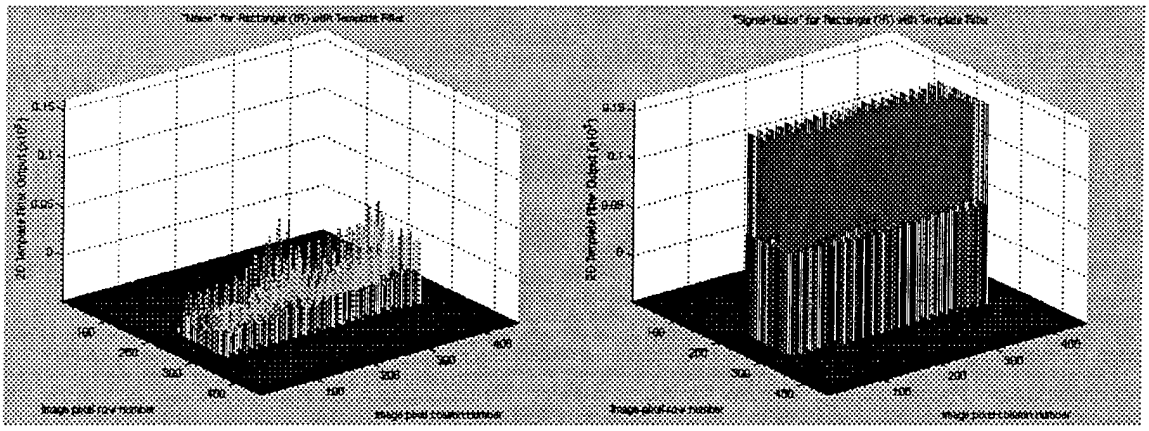


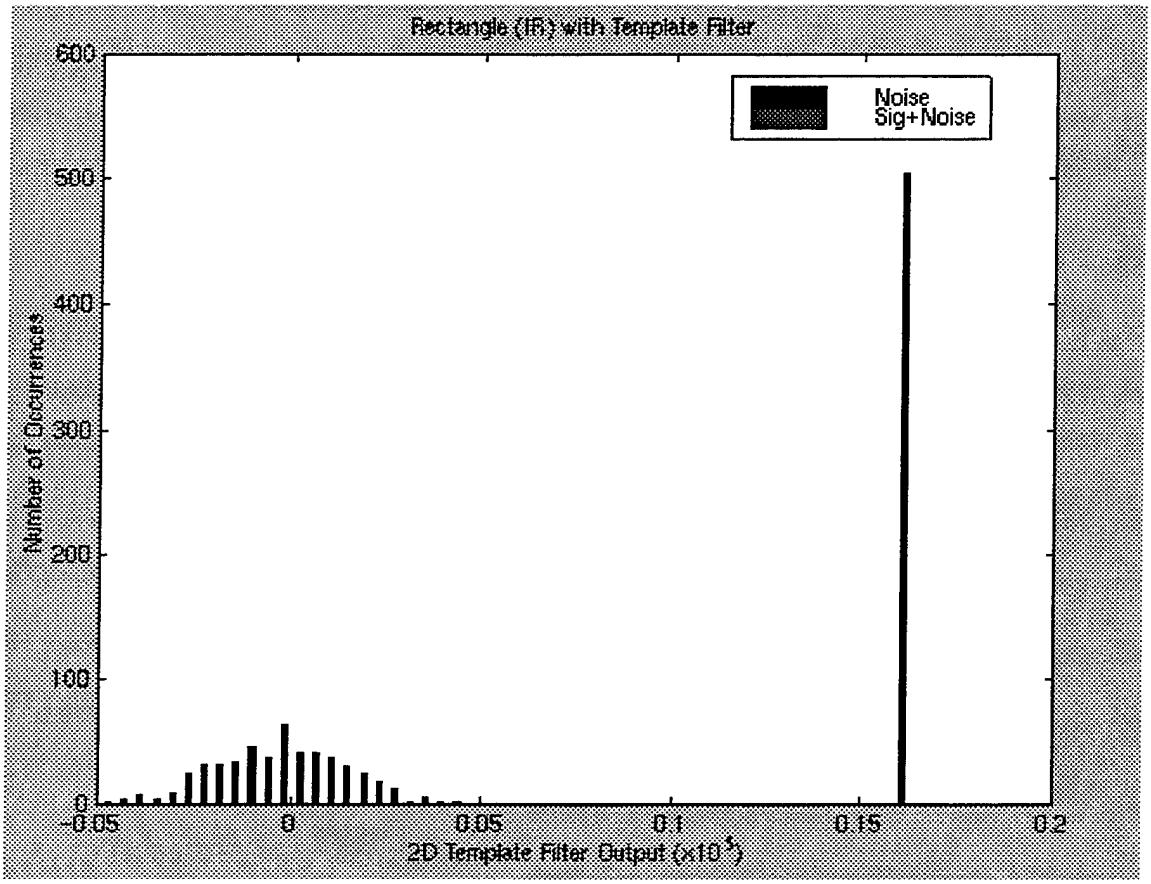


Threshold Criteria

Target Position	SNt	Nt
1	0.00125	0.000446
2	0.00189	-0.00151
3	0.00298	0.000111

"Rectangle" (IR) with Template matching filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

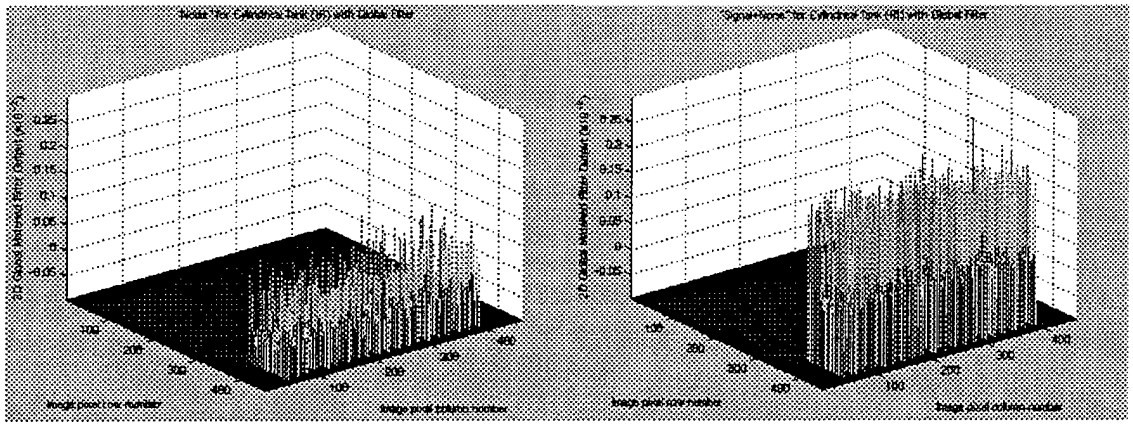


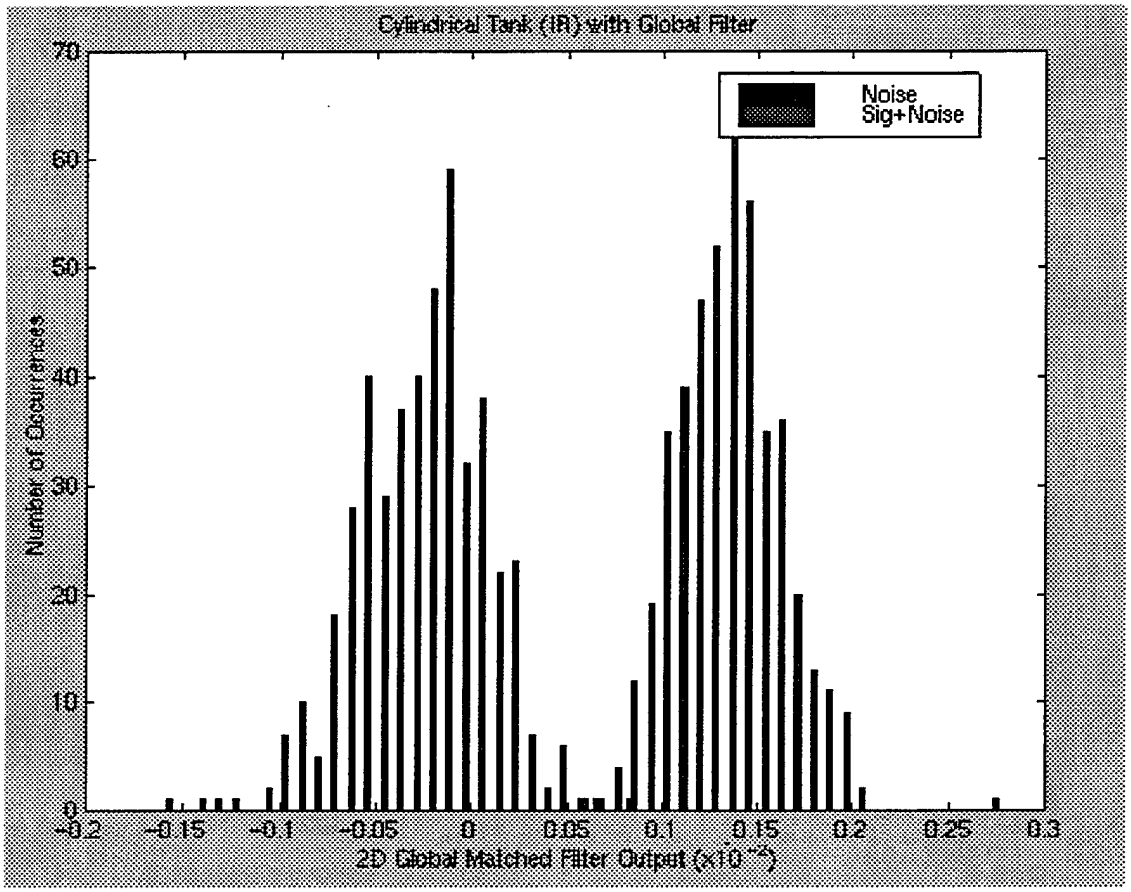


Threshold Criteria

Target Position	SNt	Nt
1	1.6e+04	-3.38e+03
2	1.6e+04	2.07e+03
3	1.6e+04	459

"Cylindrical Tank" (IR) with Global matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

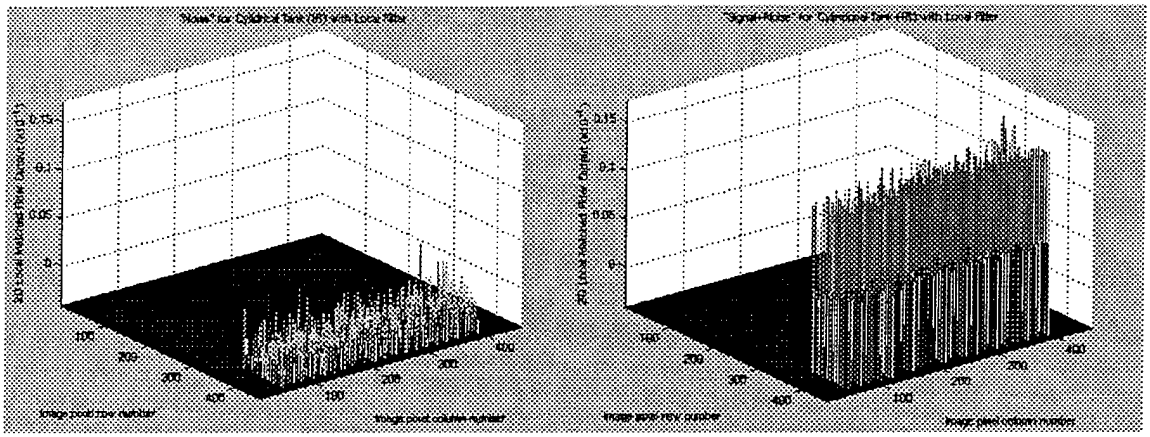


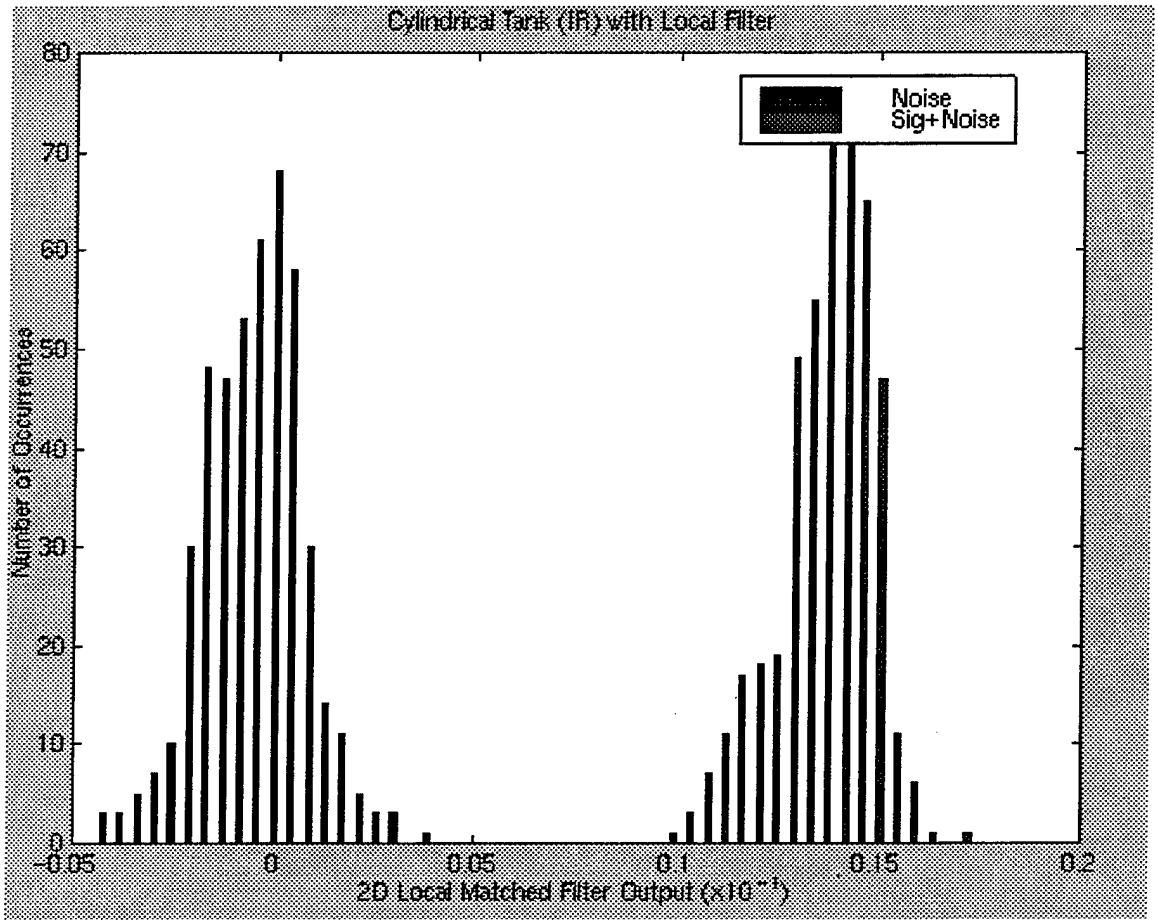


Threshold Criteria

Target Position	SNt	Nt
1	0.00135	0.000549
2	0.00123	-0.000898
3	0.00158	0.00031

"Cylindrical Tank" (IR) with Local matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

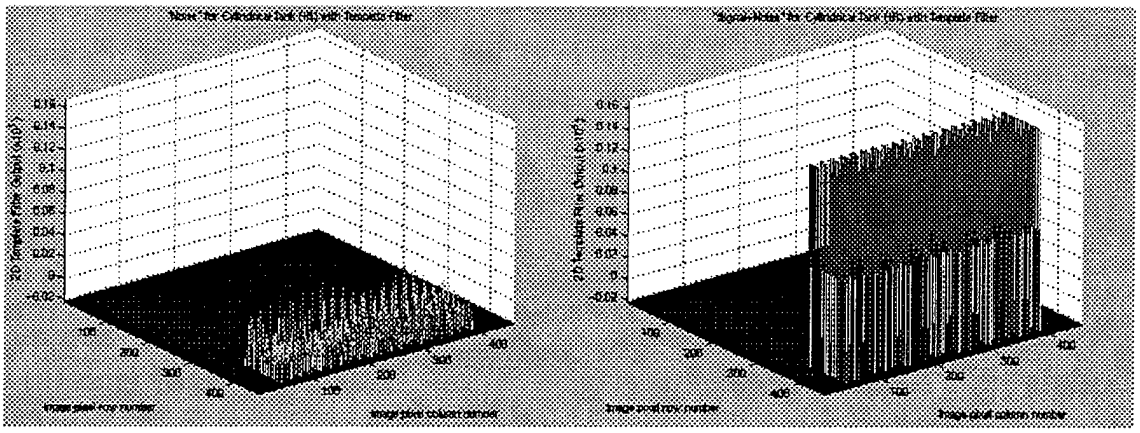


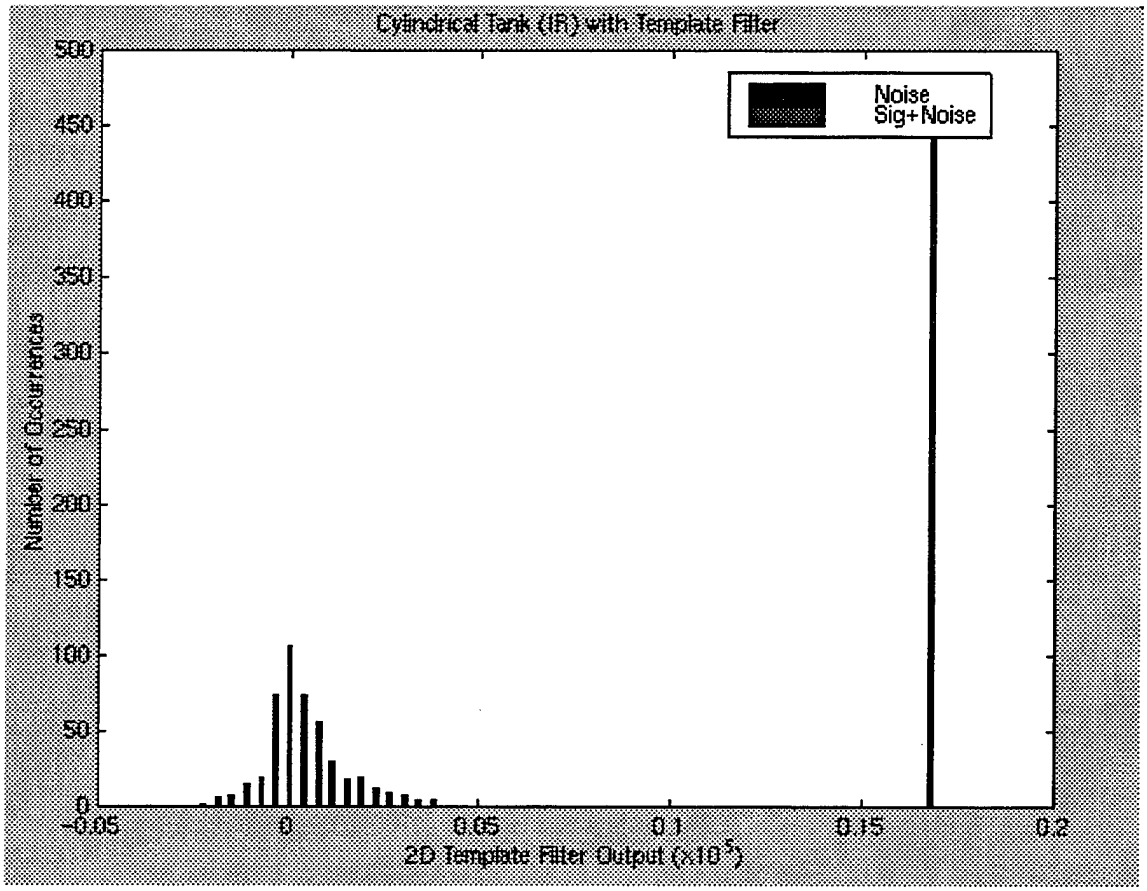


Threshold Criteria

Target Position	SNt	Nt
1	0.0132	0.00136
2	0.0124	-0.00504
3	0.0122	8.68e-05

"Cylindrical Tank" (IR) with Template matching filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

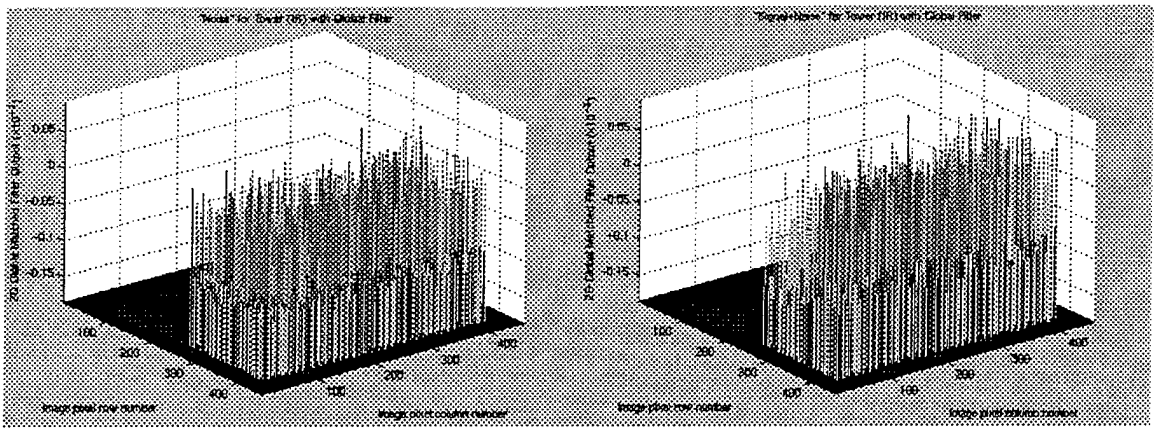


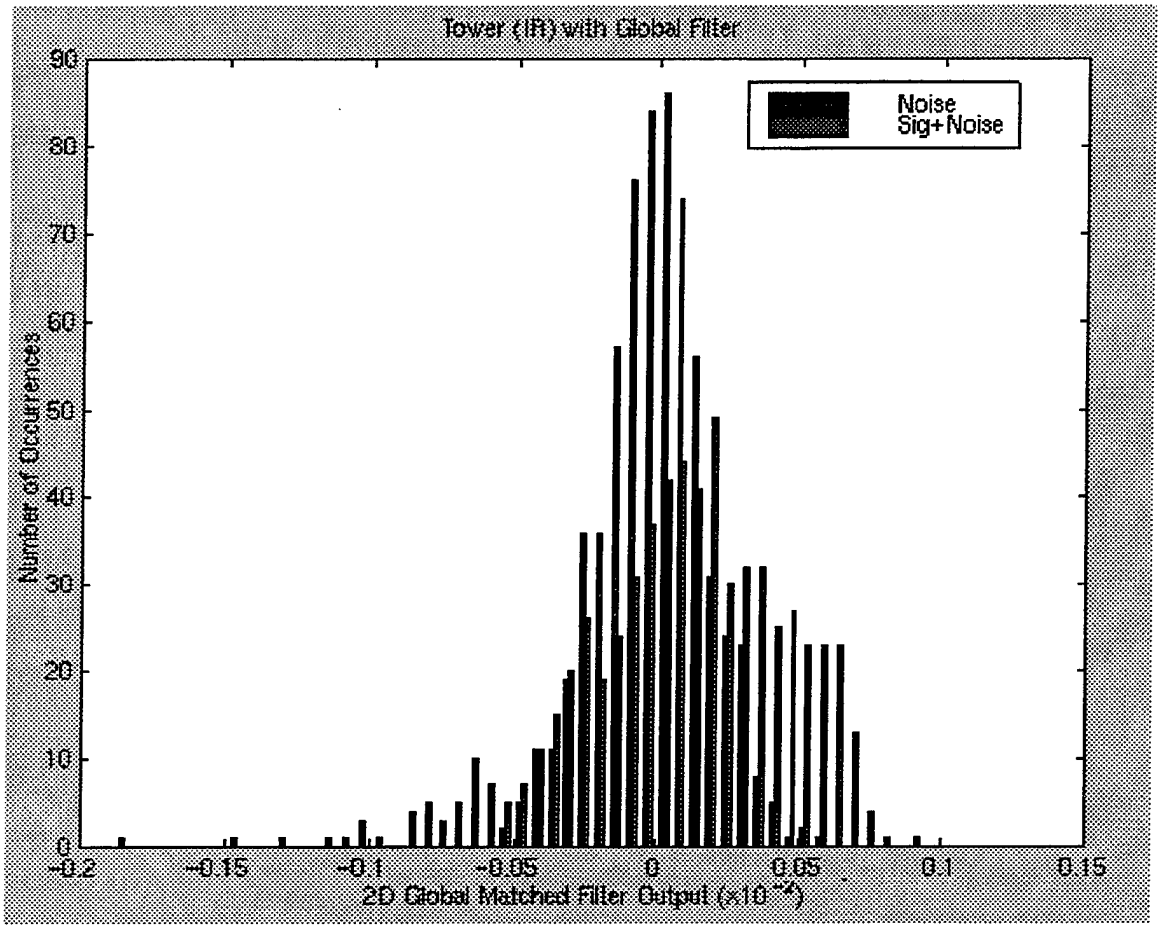


Threshold Criteria

Target Position	SNt	Nt
1	1.67e+04	1.74e+03
2	1.67e+04	816
3	1.67e+04	-1.27e+03

"Tower" (IR) with Global matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

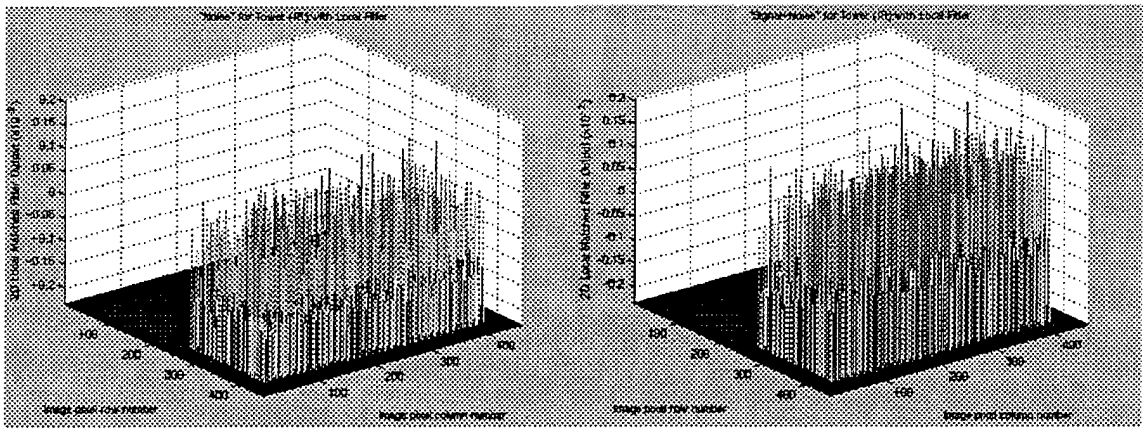


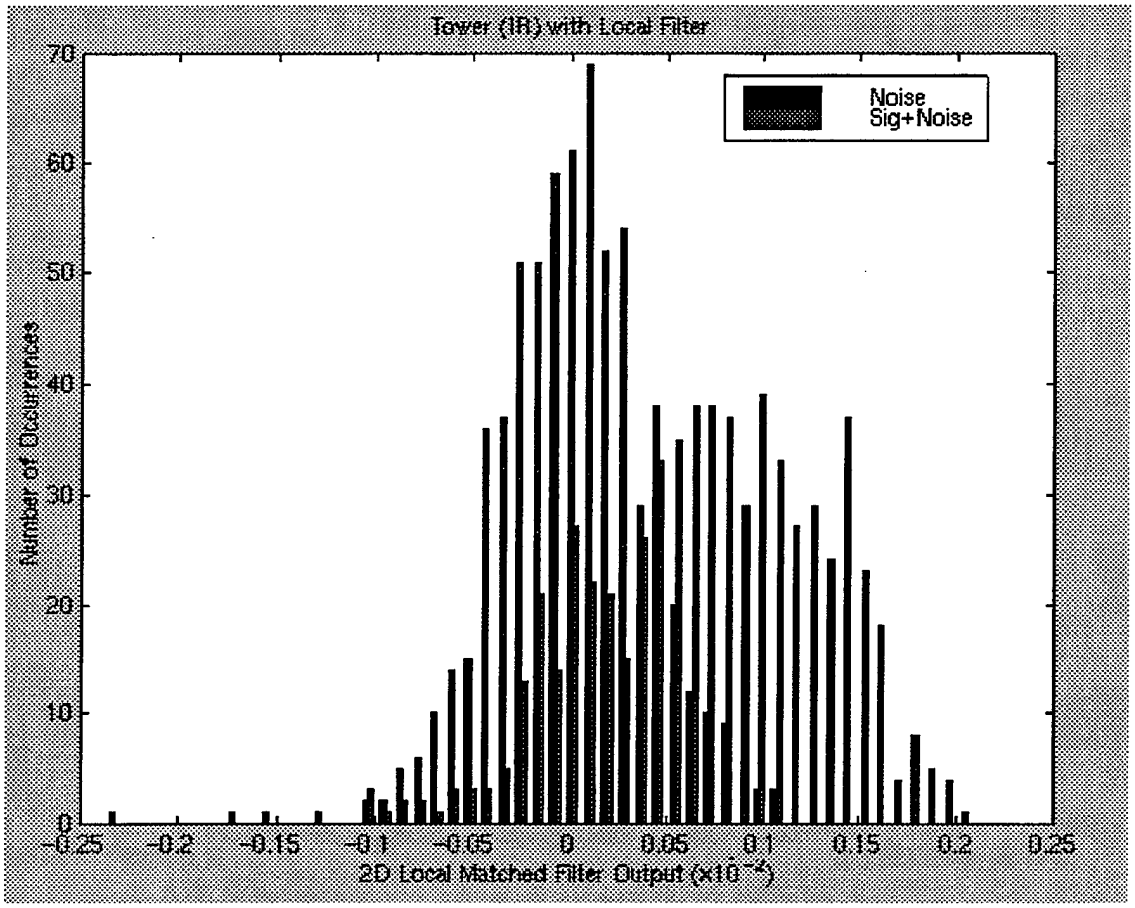


Threshold Criteria

Target Position	SNt	Nt
1	0.000216	-0.000125
2	-4.63e-05	0.000211
3	0.000248	3.19e-05

"Tower" (IR) with Local matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

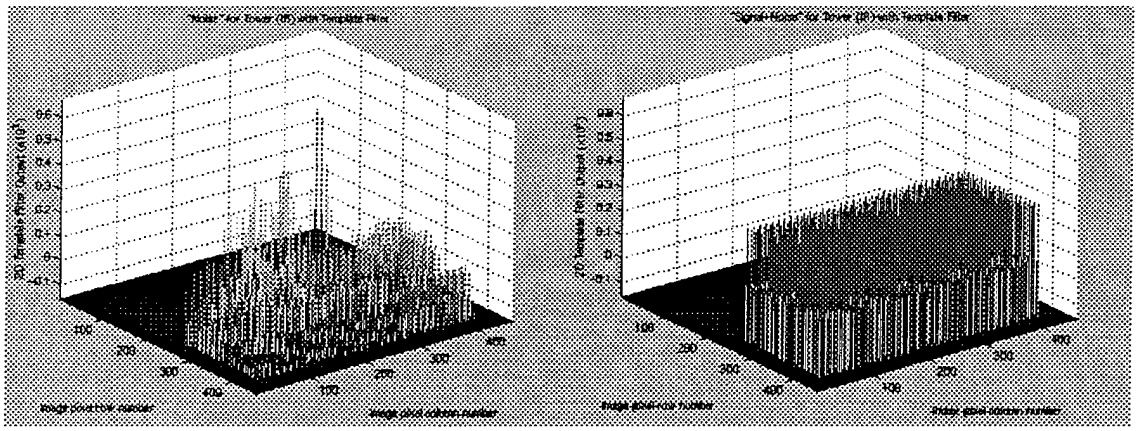


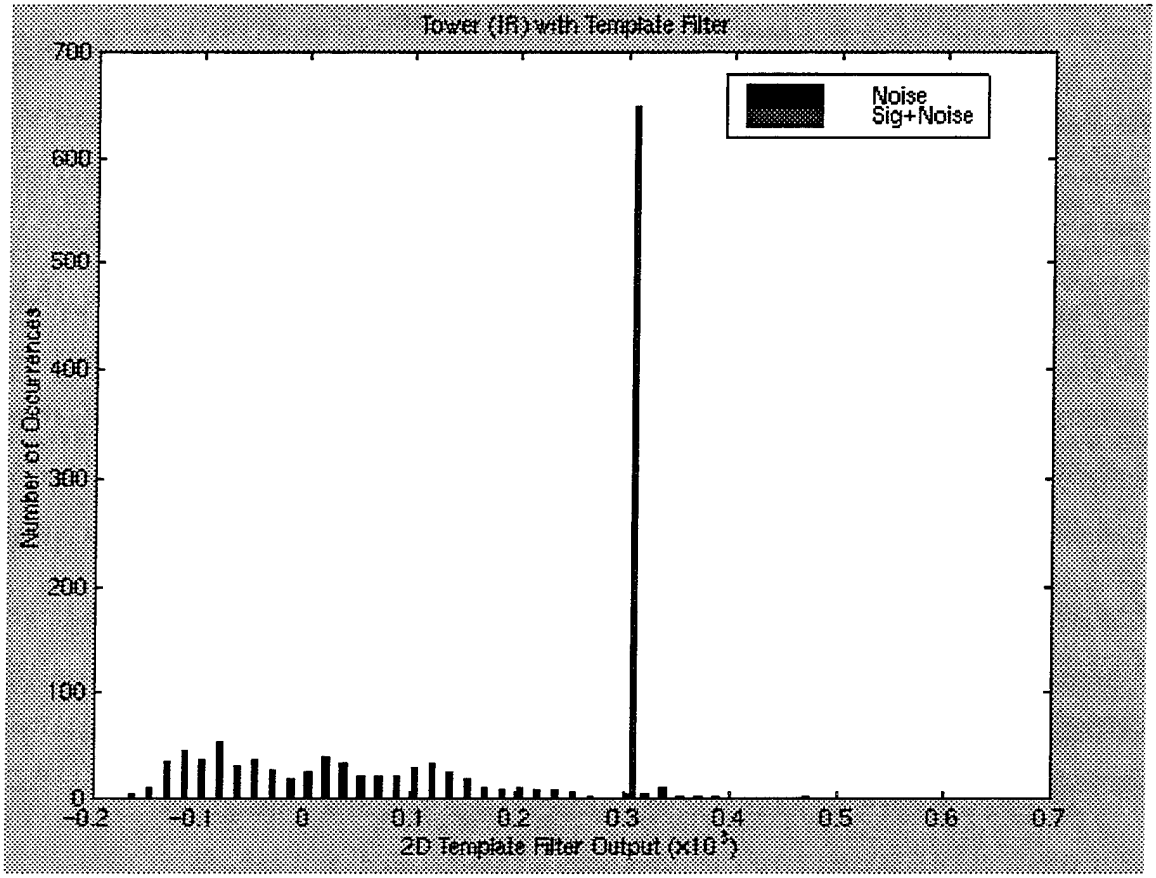


Threshold Criteria

Target Position	SNt	Nt
1	0.00107	0.000233
2	0.000752	3.51e-05
3	0.000337	-0.000686

"Tower" (IR) with Template matching filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

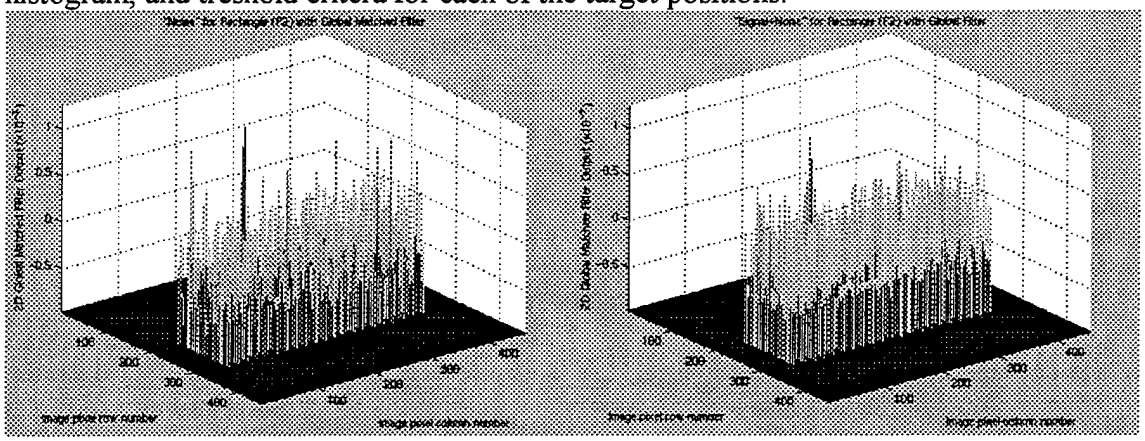


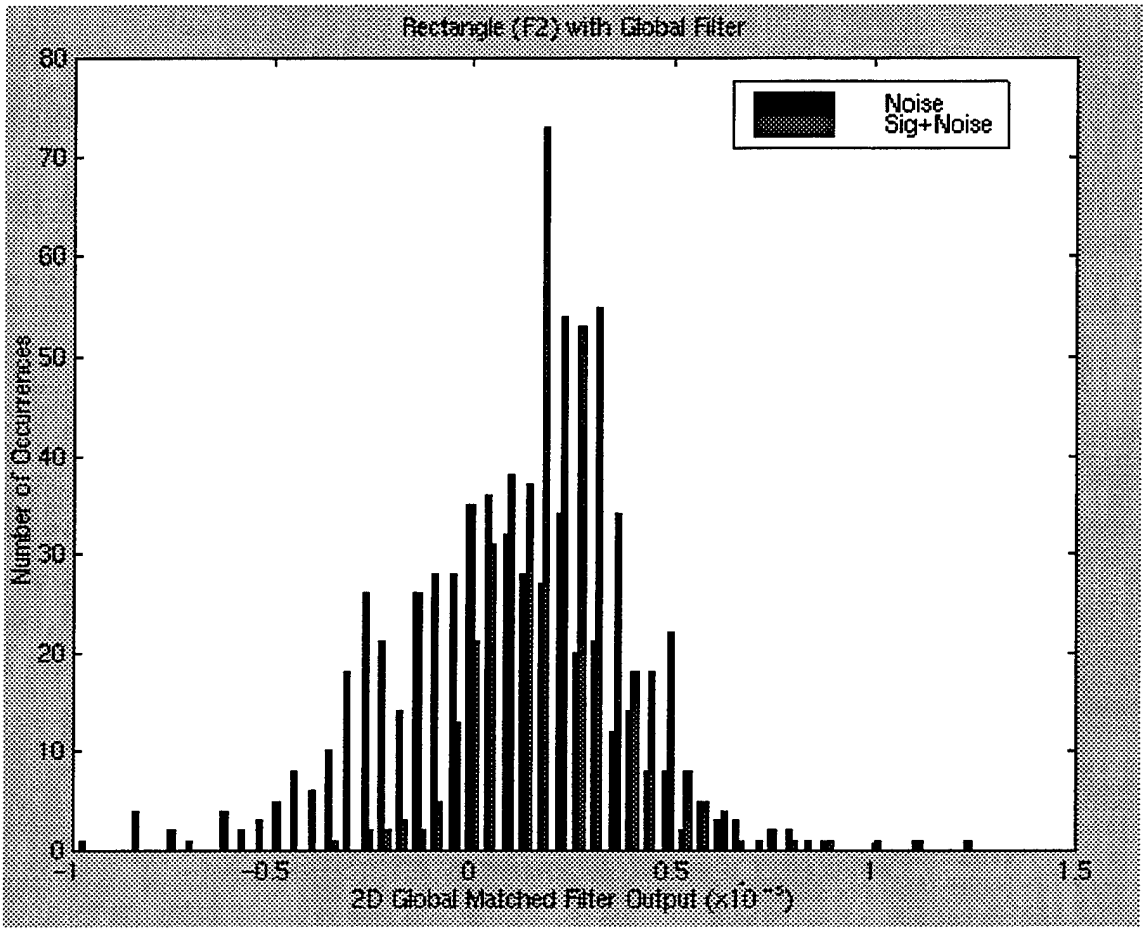


Threshold Criteria

Target Position	SNt	Nt
1	310	58.9
2	310	-40.4
3	310	30.4

"Rectangle" (F2) with Global matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

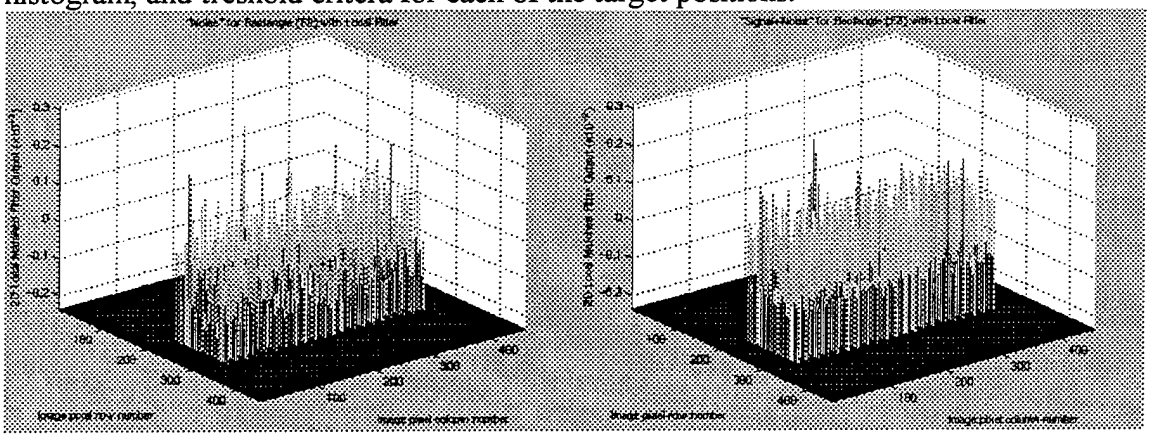


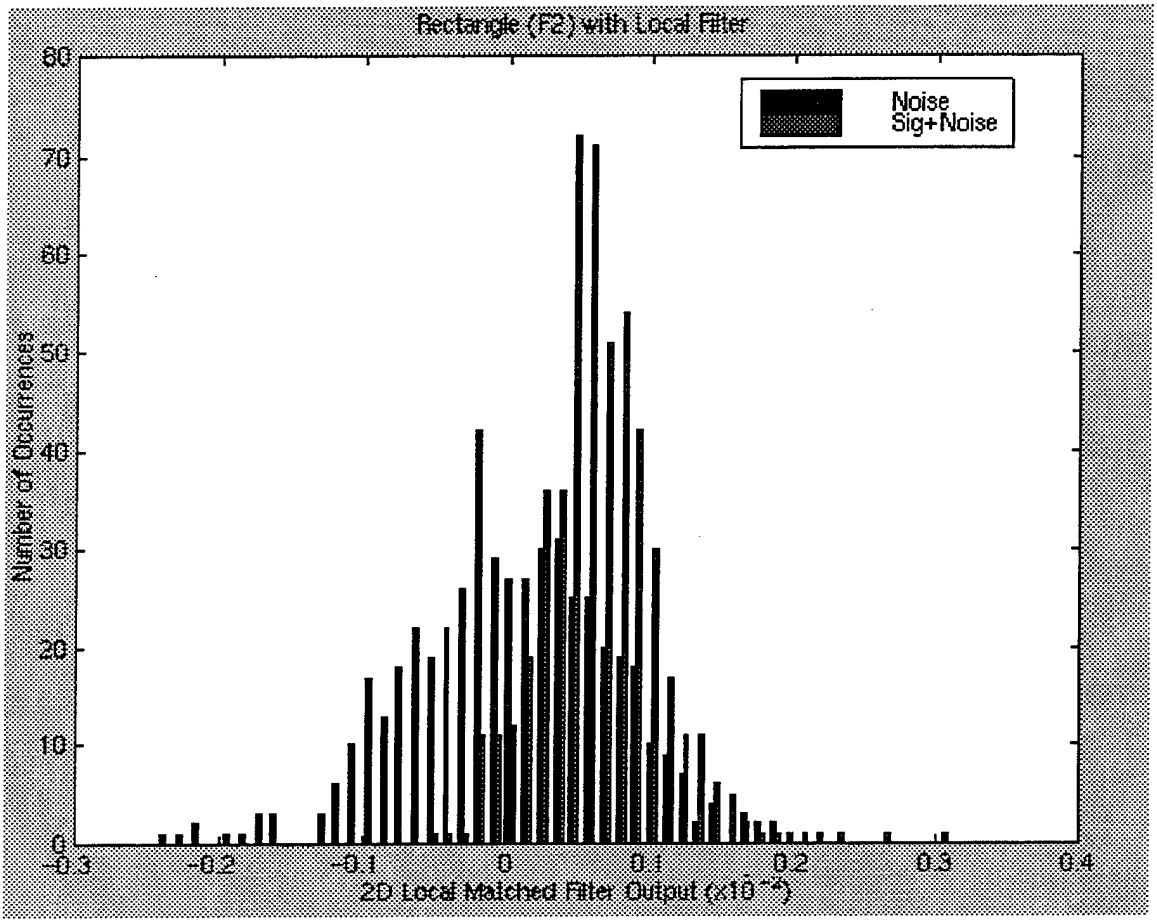


Threshold Criteria

Target Position	SNt	Nt
1	0.000192	0.000366
2	0.000164	0.0002
3	0.000317	7.99e-05

"Rectangle" (F2) with Local matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

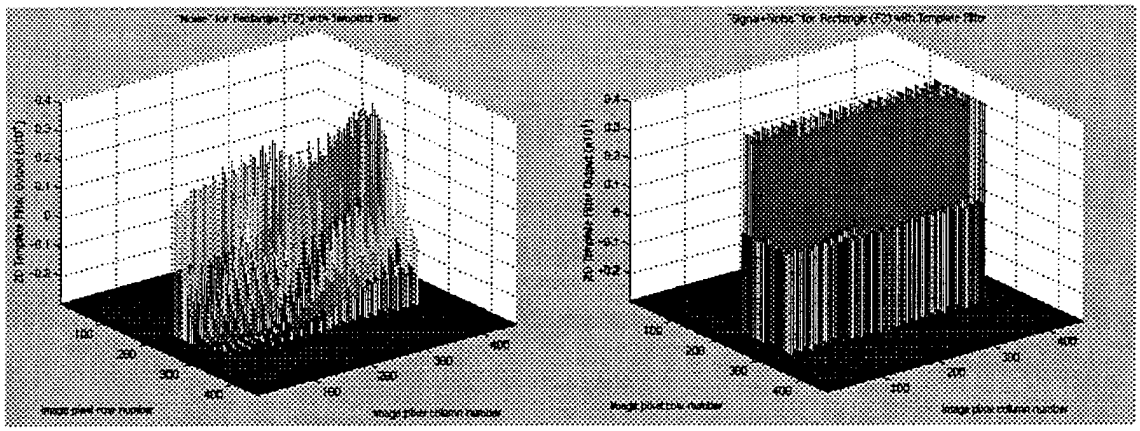


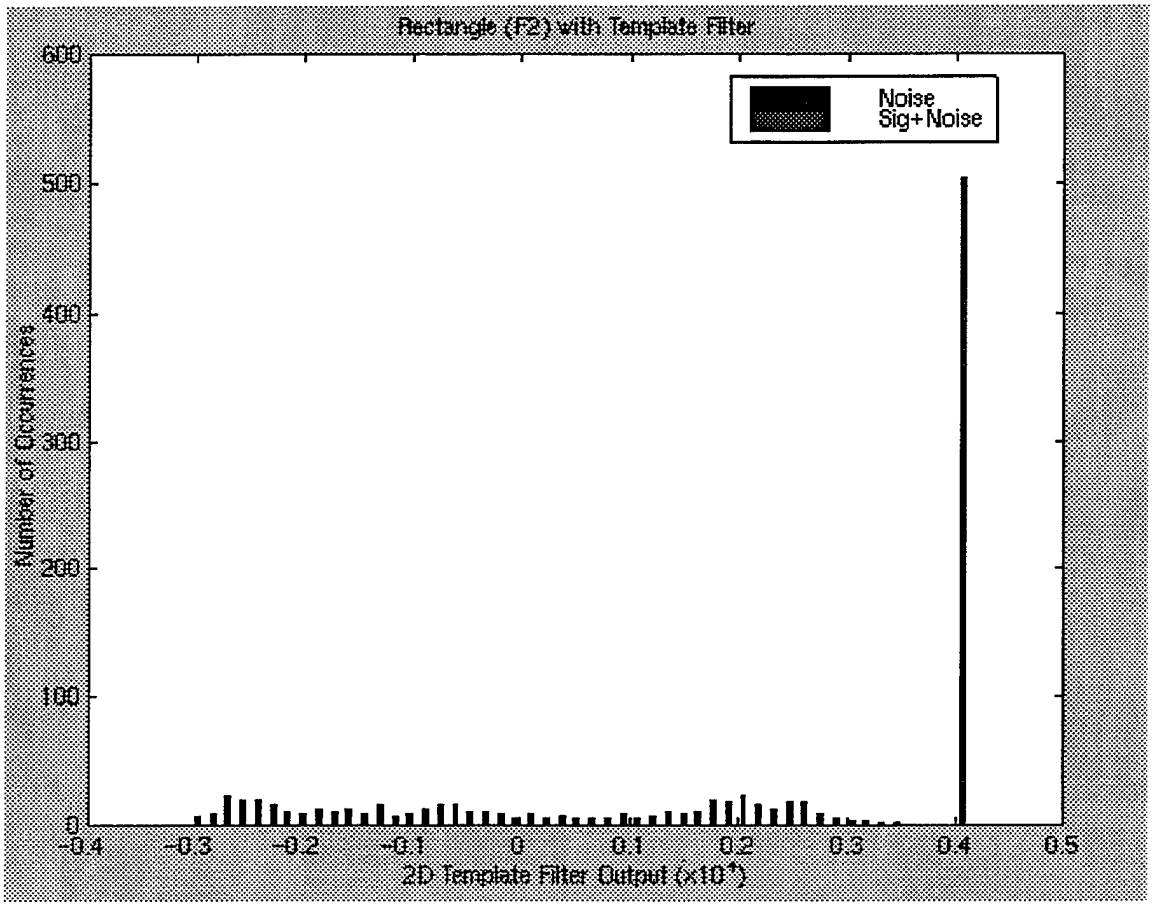


Threshold Criteria

Target Position	SNt	Nt
1	0.000382	0.000993
2	0.000554	0.00119
3	0.00079	0.000185

"Rectangle" (F2) with Template matching filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

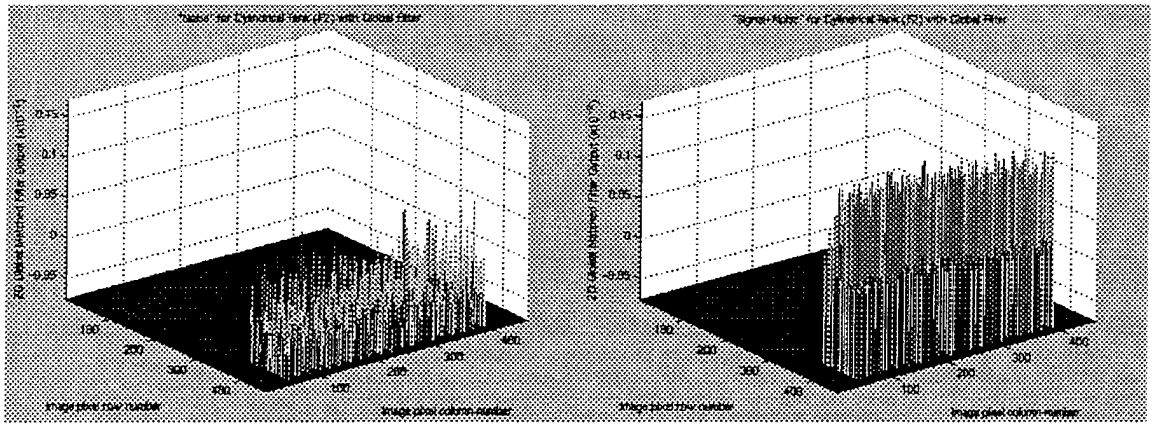


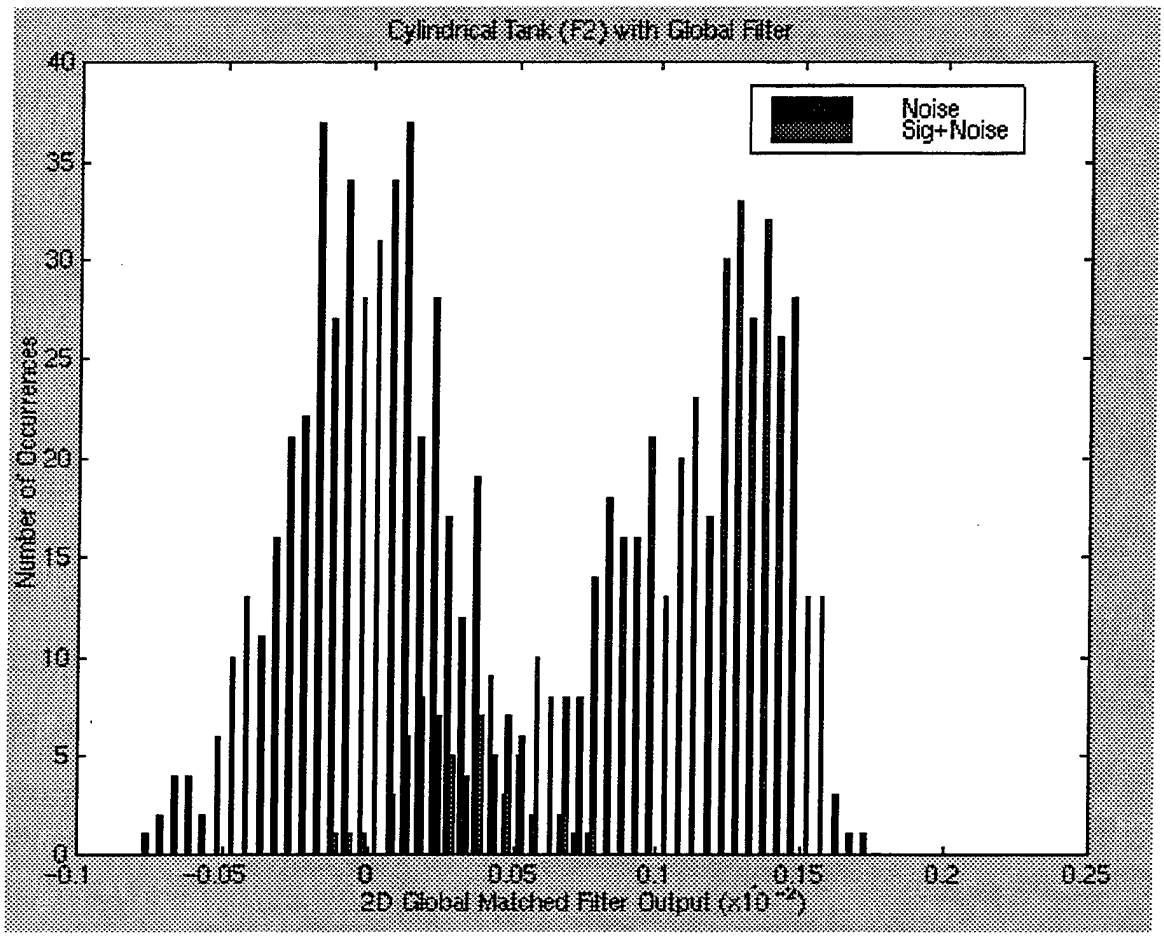


Threshold Criteria

Target Position	SNt	Nt
1	4.04e+03	1.46e+03
2	4.04e+03	2.46e+03
3	4.04e+03	2.47e+03

"Cylindrical Tank" (F2) with Global matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

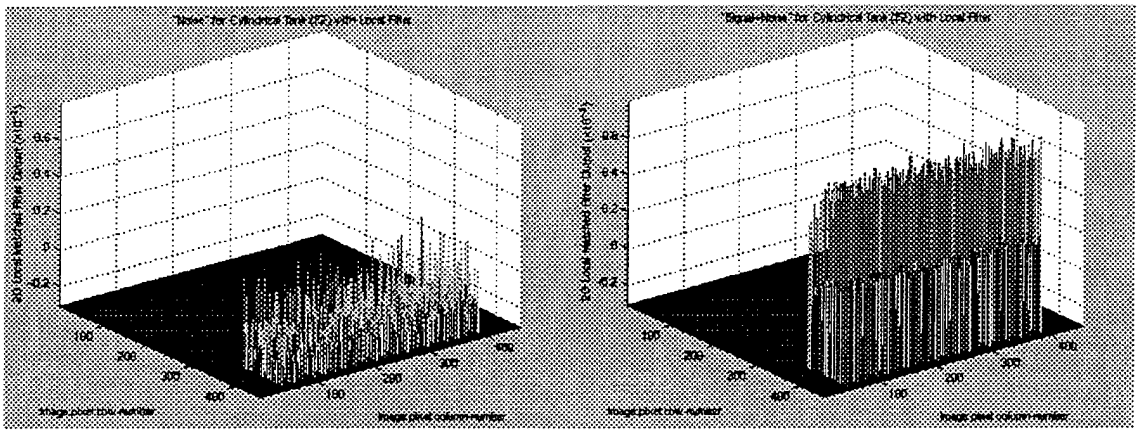


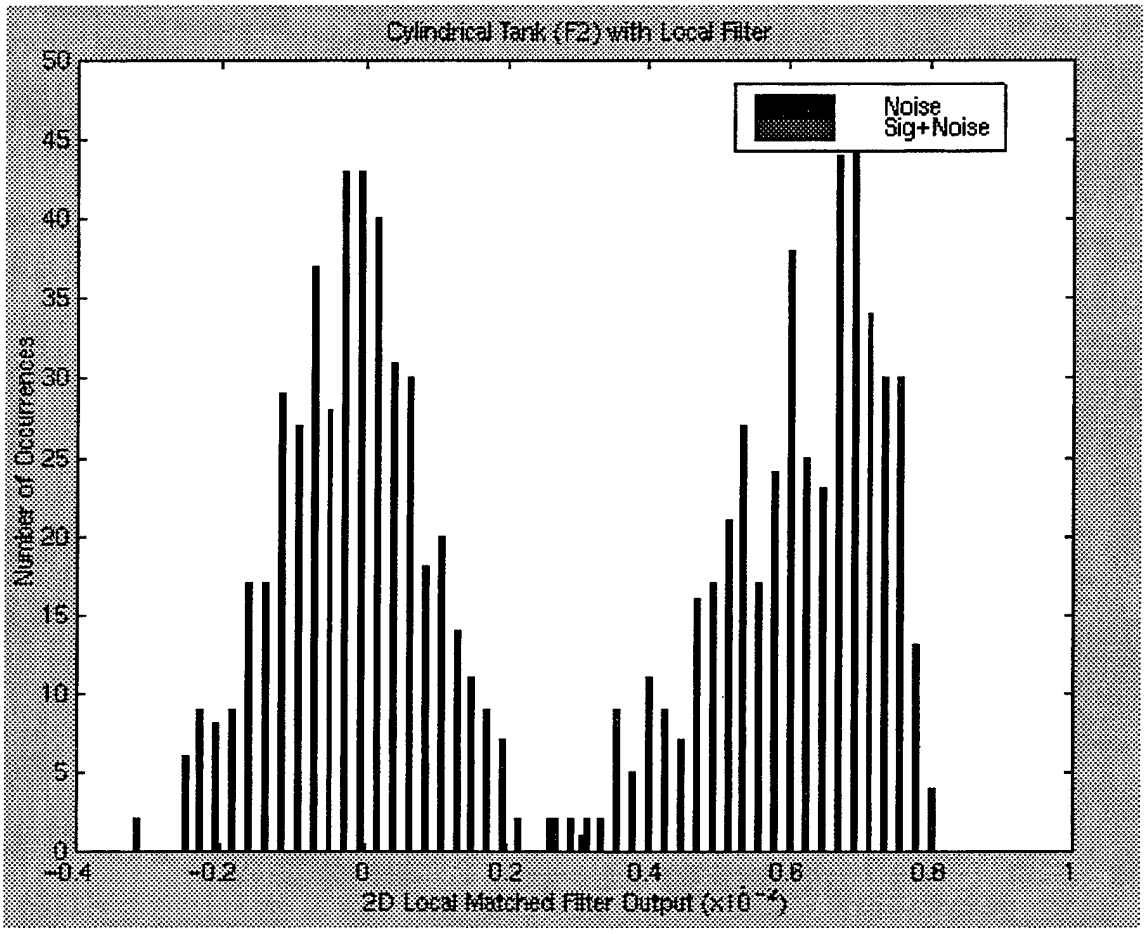


Threshold Criteria

Target Position	SNt	Nt
1	0.000295	-0.000349
2	0.00057	6.55e-05
3	0.000224	-0.000157

"Cylindrical Tank" (F2) with Local matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

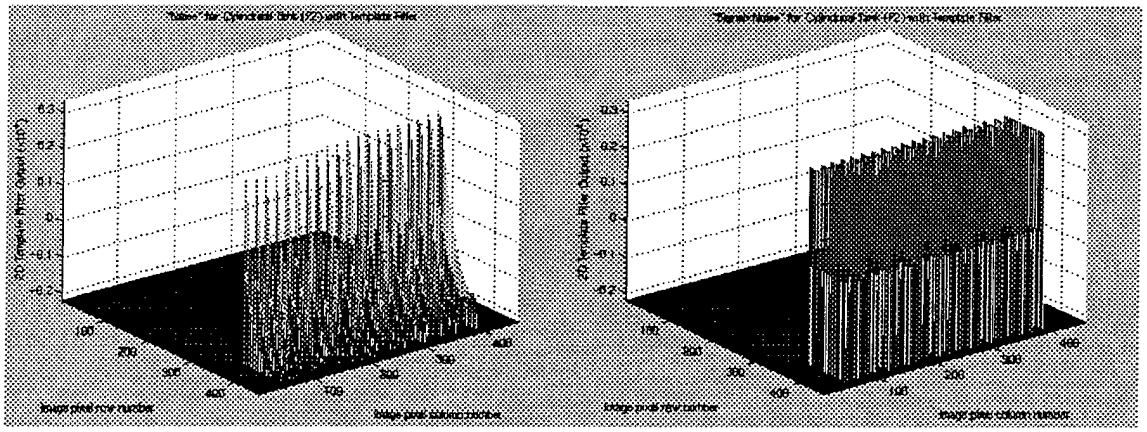


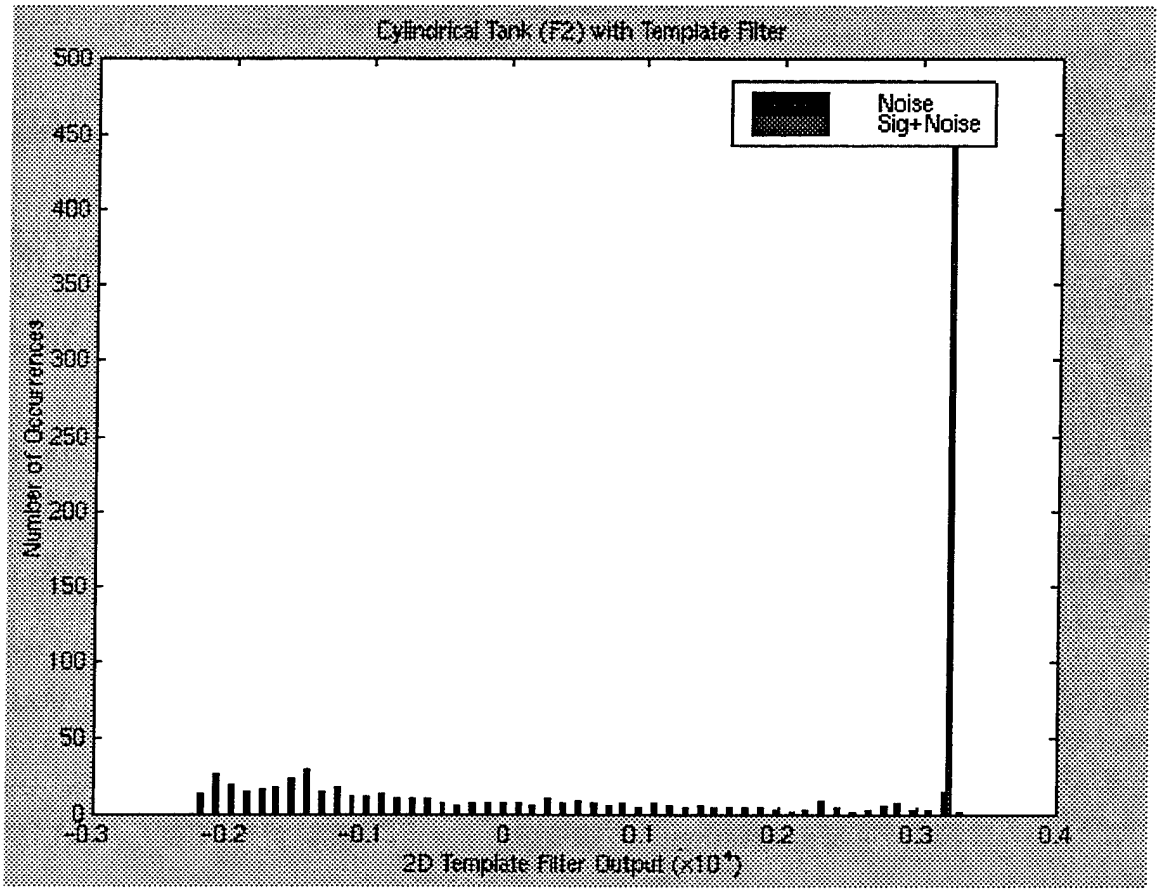


Threshold Criteria

Target Position	SNt	Nt
1	0.00434	-0.000187
2	0.00504	0.000649
3	0.00371	-0.000196

"Cylindrical Tank" (F2) with Template matching filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

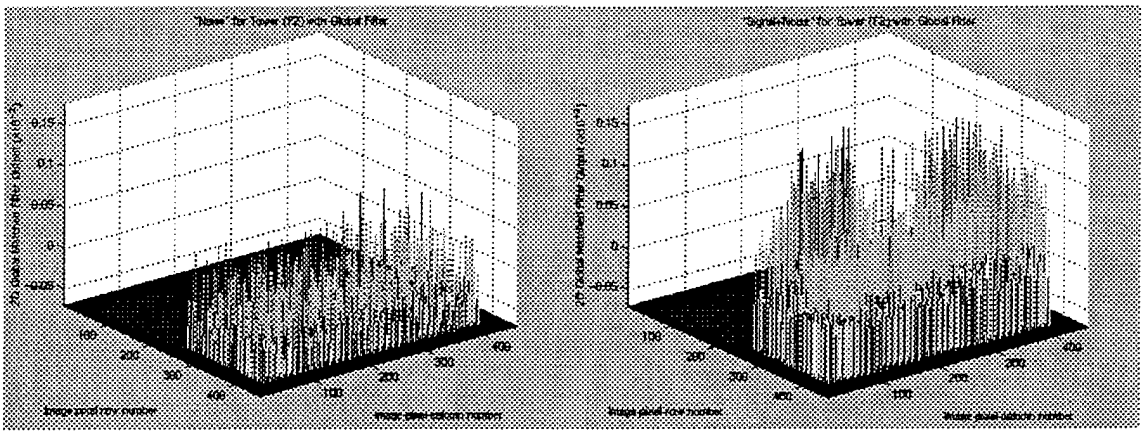


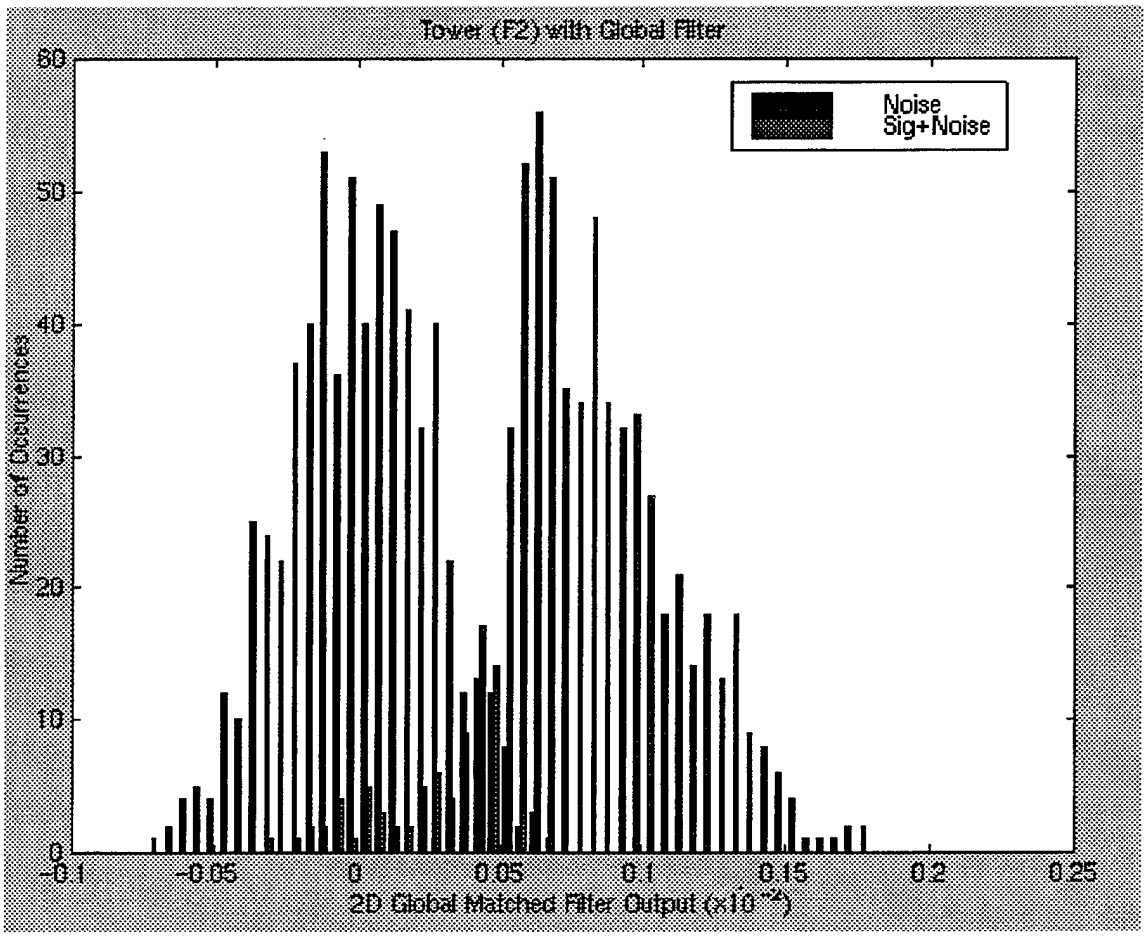


Threshold Criteria

Target Position	SNt	Nt
1	3.22e+03	2.79e+03
2	3.22e+03	2.77e+03
3	3.22e+03	2.84e+03

"Tower" (F2) with Global matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

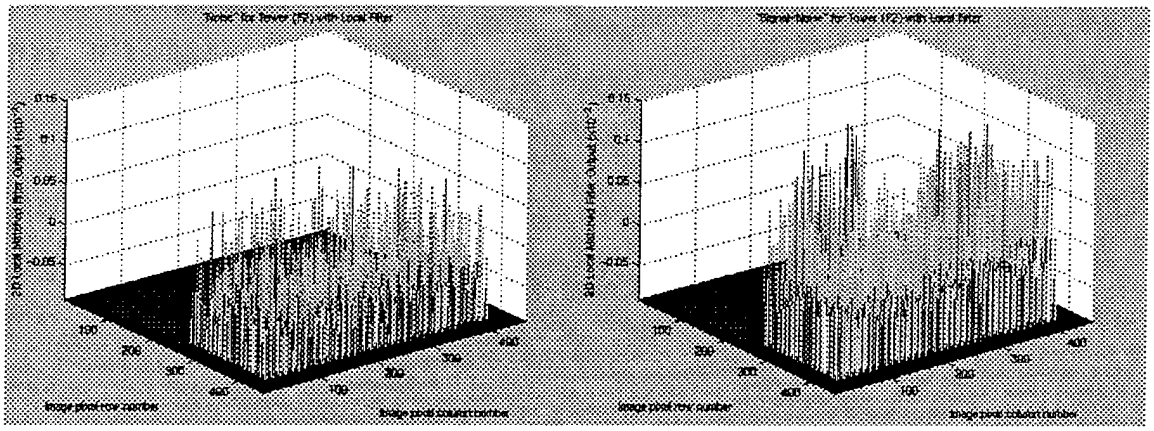


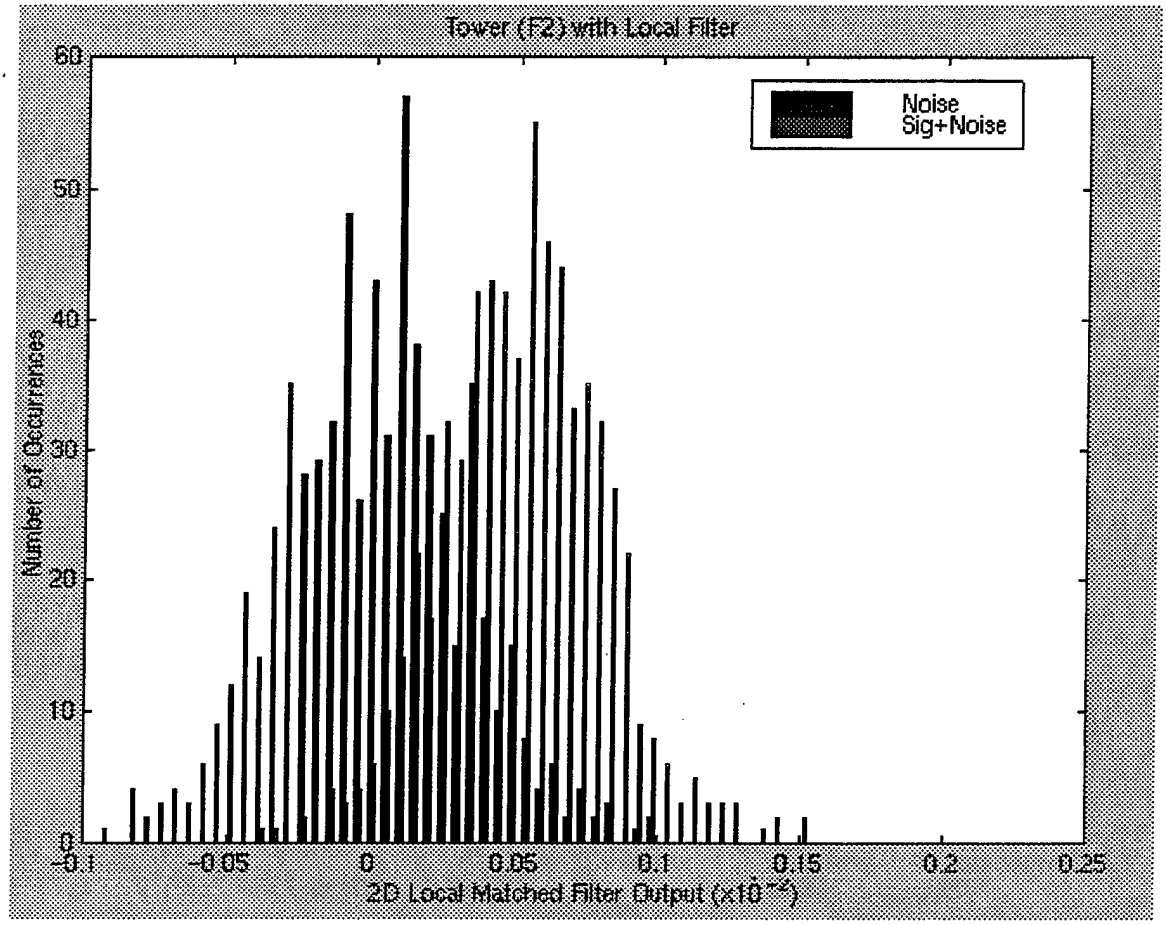


Threshold Criteria

Target Position	SNt	Nt
1	0.000476	-0.000111
2	0.000624	0.000421
3	0.000824	2.27e-05

"Tower" (F2) with Local matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

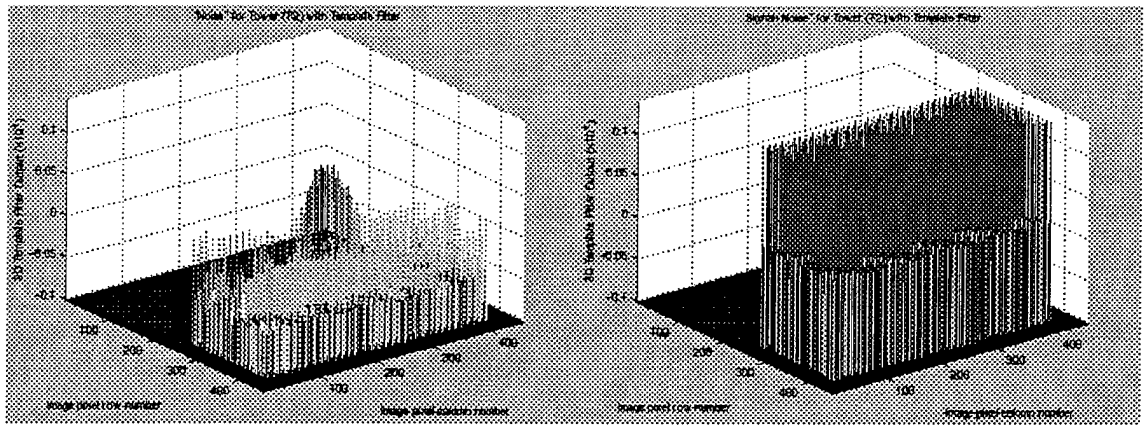


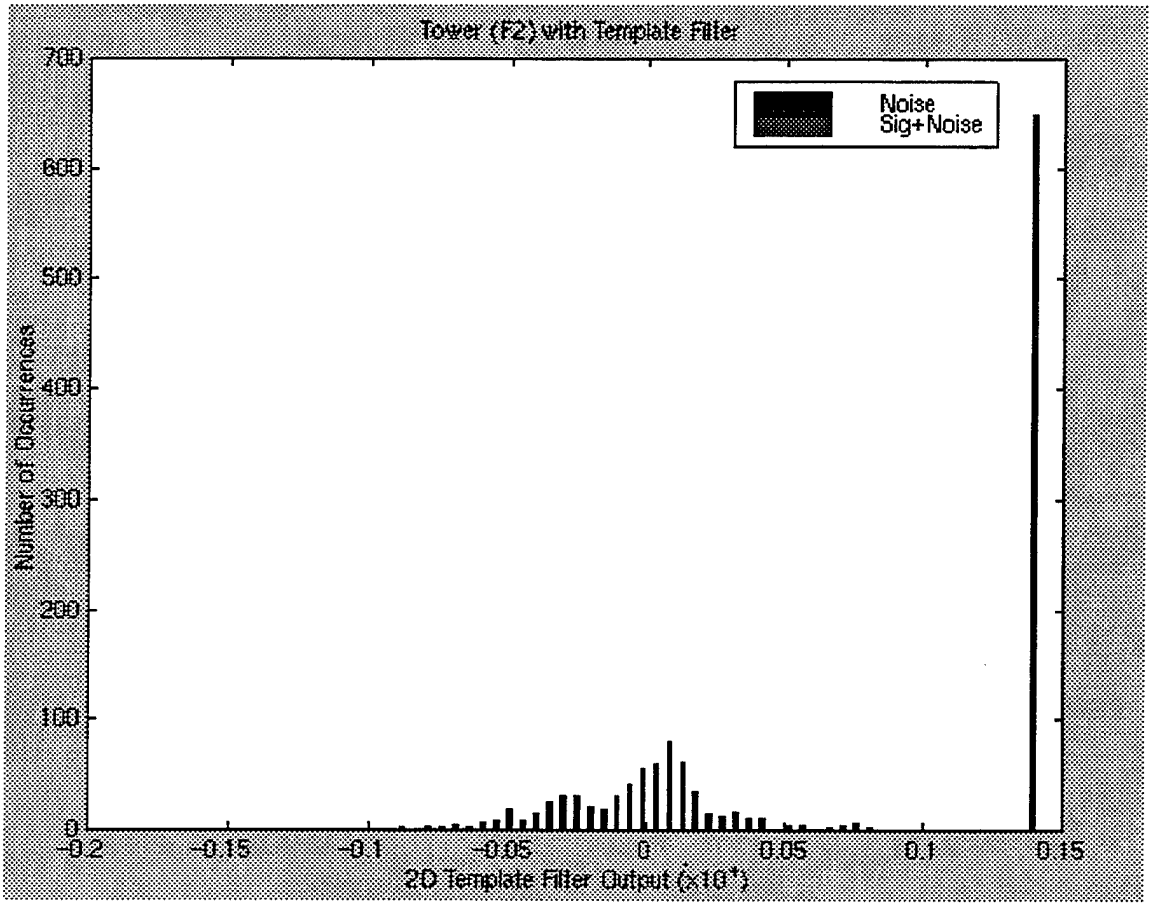


Threshold Criteria

Target Position	SNt	Nt
1	0.000561	0.000327
2	0.00042	-0.000461
3	0.000576	0.000233

"Tower" (F2) with Template matching filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

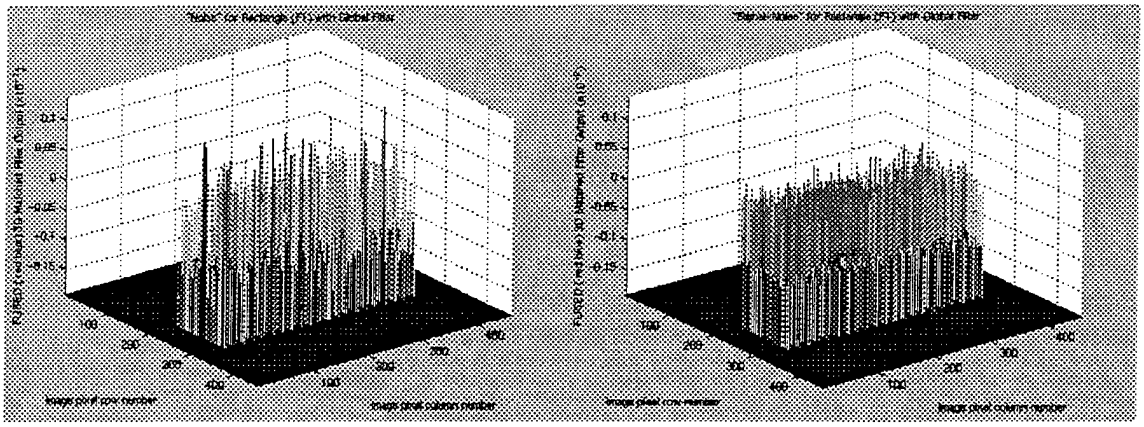


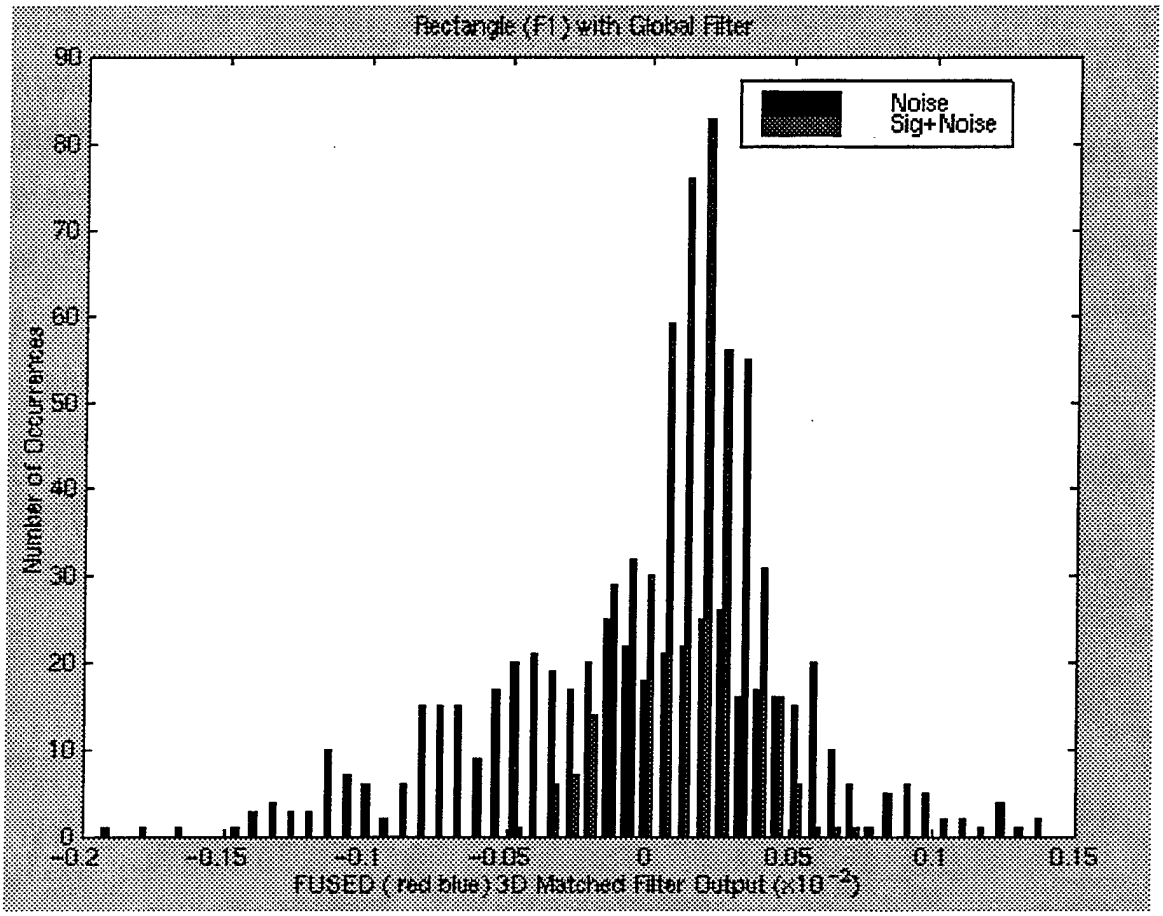


Threshold Criteria

Target Position	SNt	Nt
1	1.39e+03	133
2	1.39e+03	-73.6
3	1.39e+03	48.4

"Rectangle" (F1) with Global matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

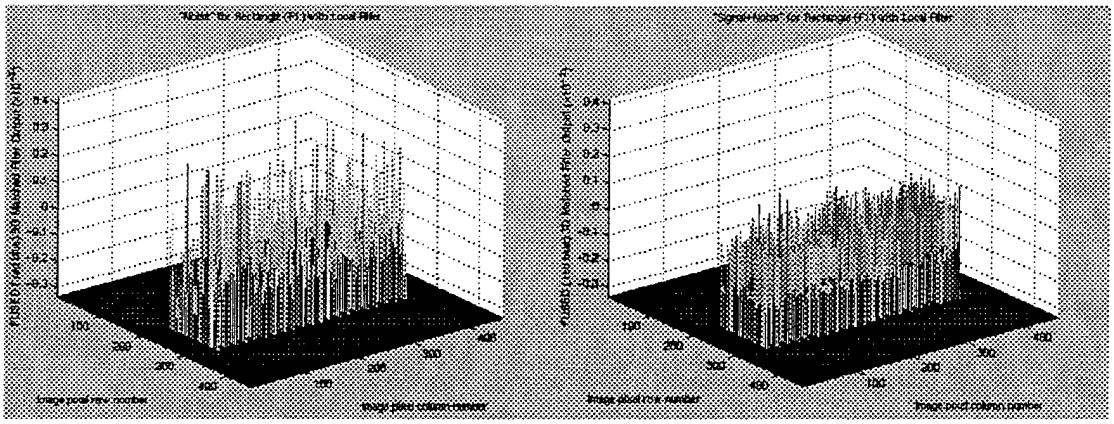


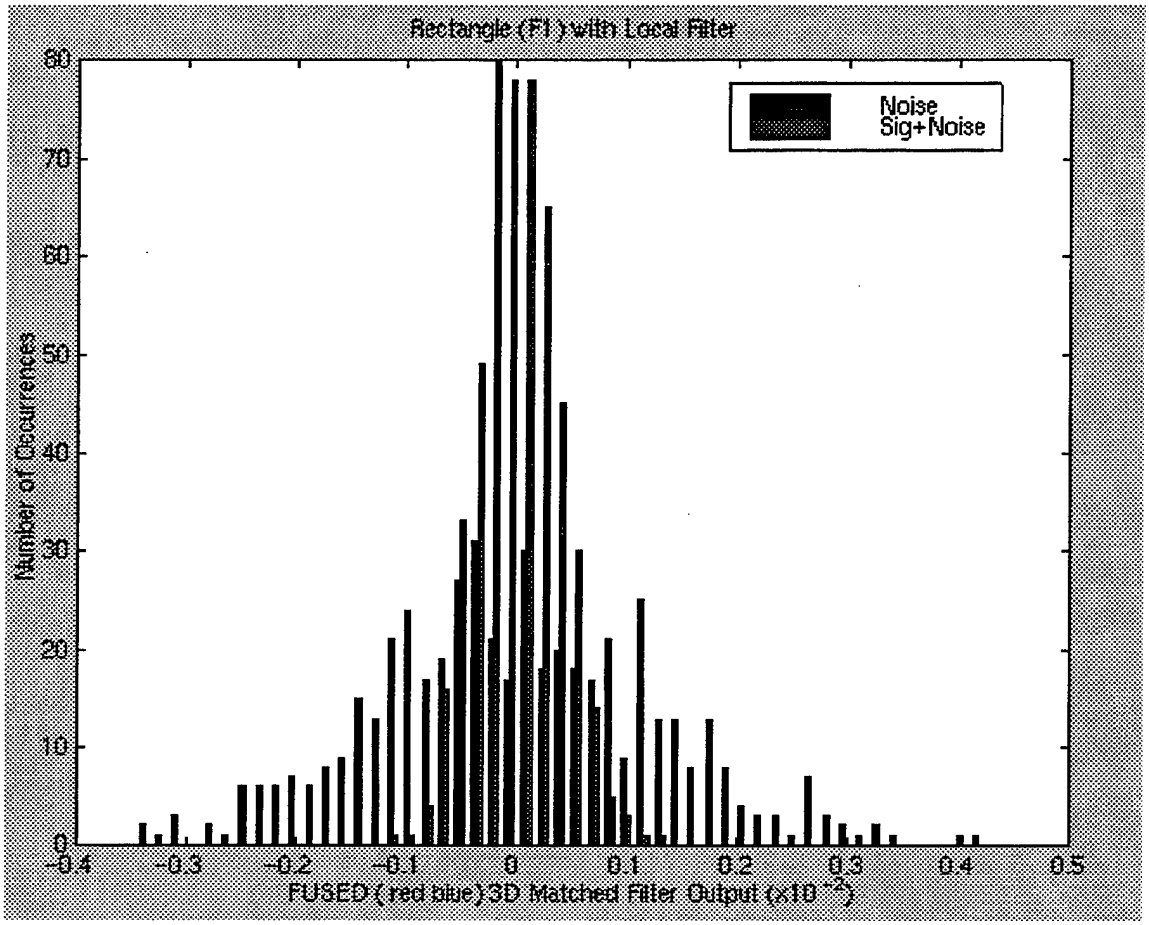


Threshold Criteria

Target Position	SNt	Nt
1	0.000394	-0.000587
2	4.27e-05	0.0002
3	-0.000178	-0.000493

"Rectangle" (F1) with Local matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

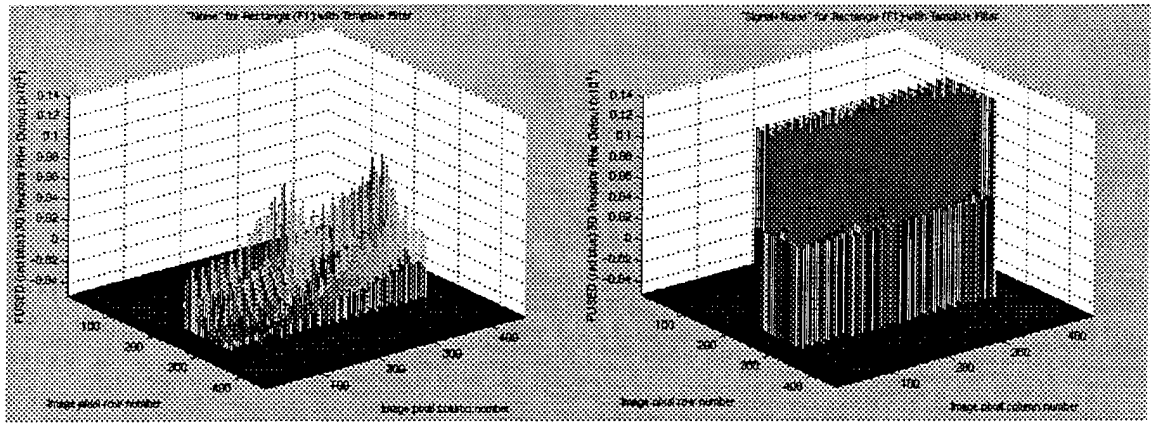


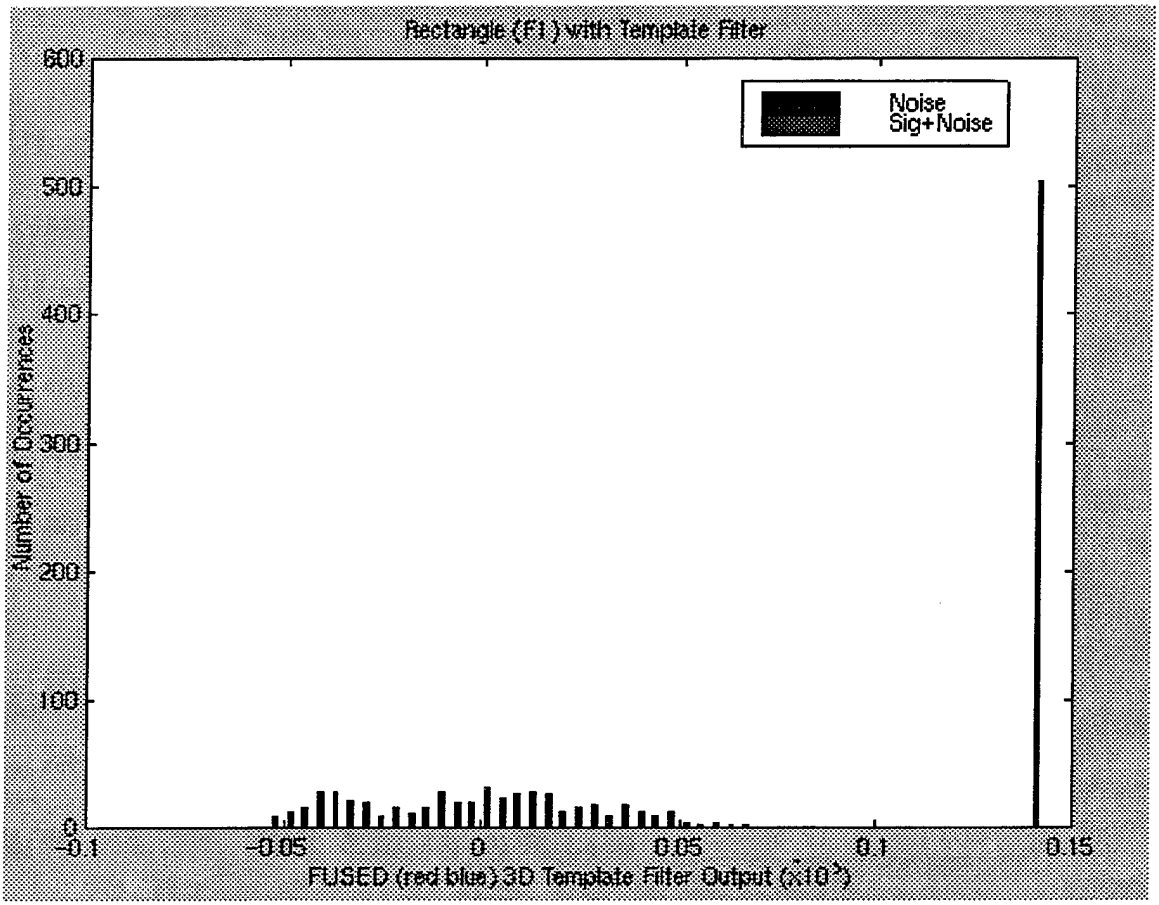


Threshold Criteria

Target Position	SNt	Nt
1	0.000678	-0.000281
2	-0.000195	0.00332
3	-0.00065	0.000916

"Rectangle" (F1) with Template matching filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

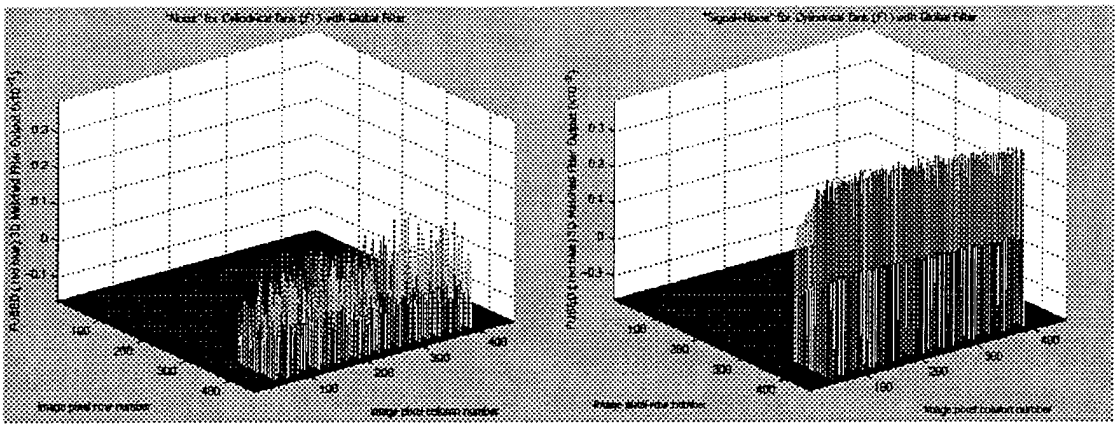


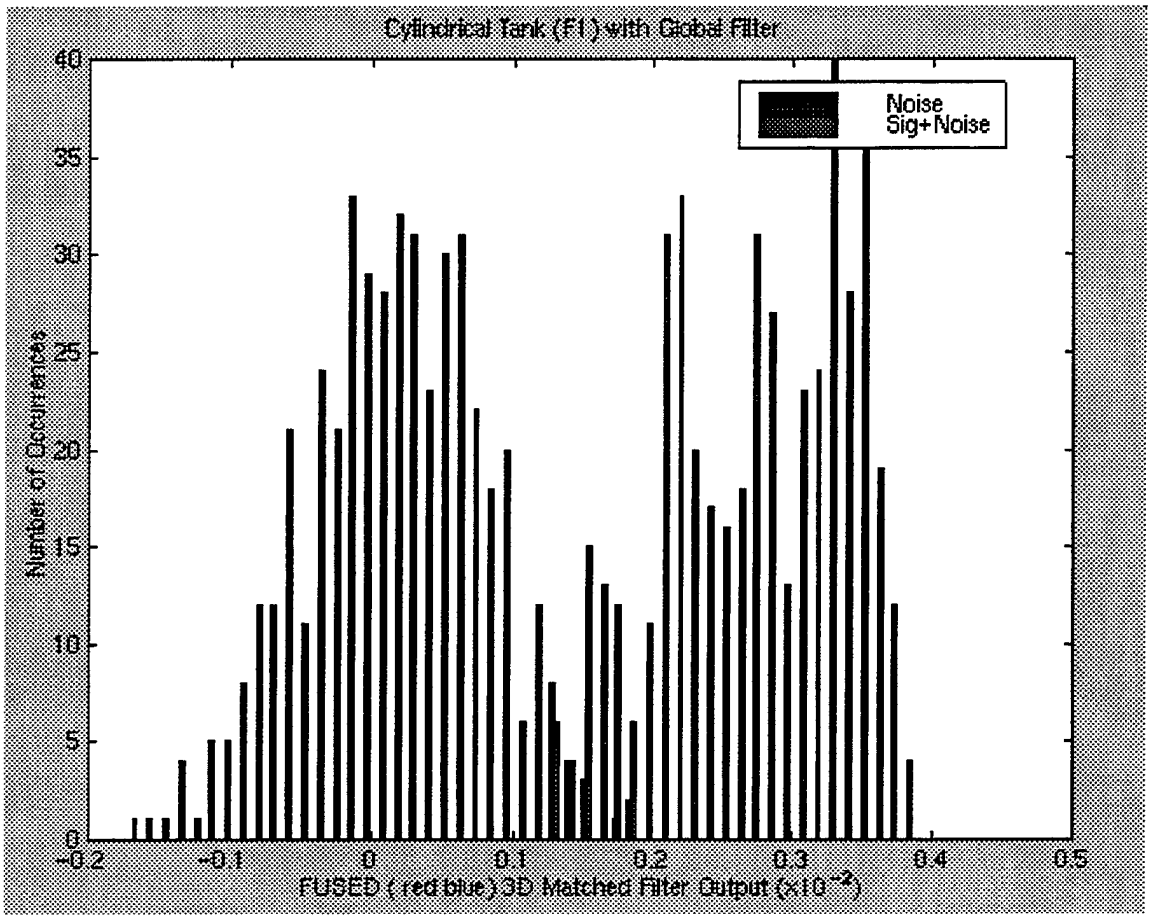


Threshold Criteria

Target Position	SNt	Nt
1	1.41e+04	-322
2	1.41e+04	4.73e+03
3	1.41e+04	3.39e+03

"Cylindrical Tank" (F1) with Global matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

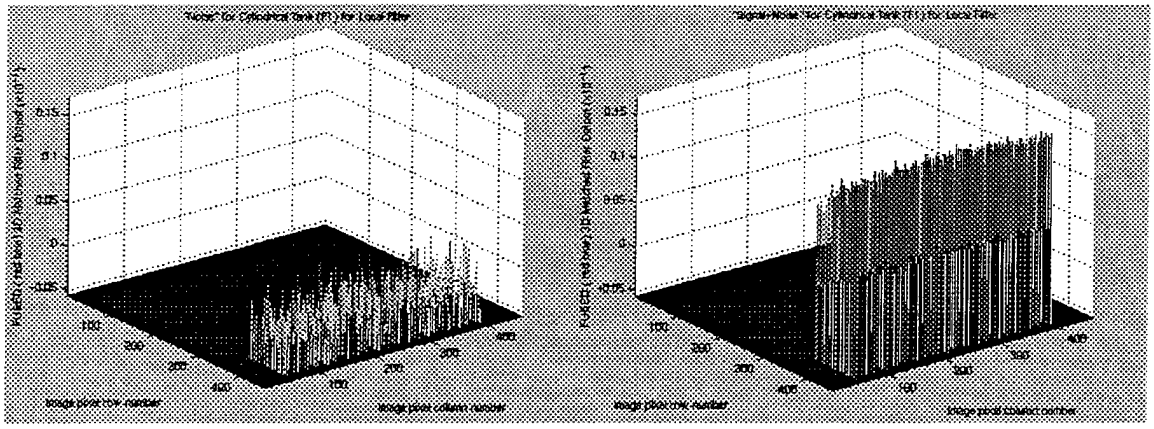


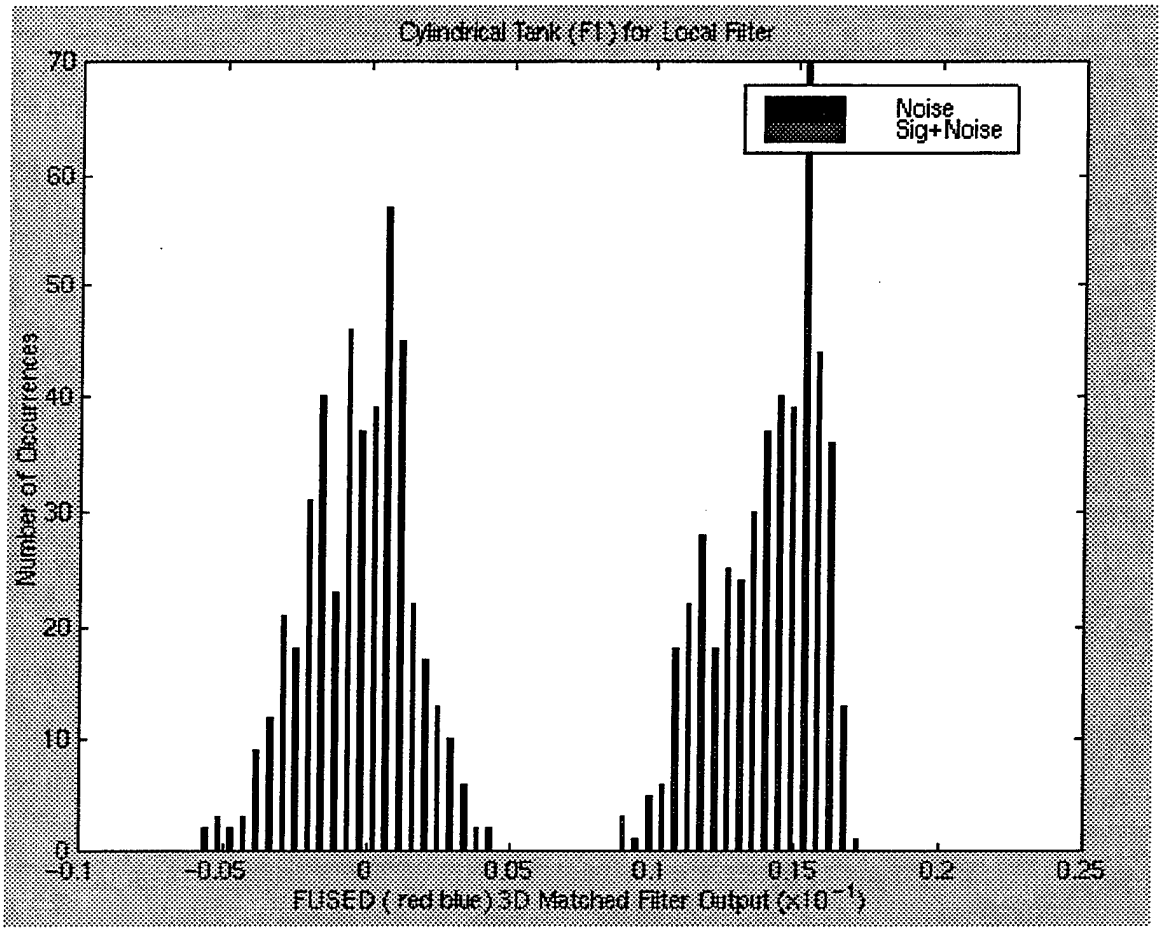


Threshold Criteria

Target Position	SNt	Nt
1	0.00179	0.000238
2	0.00163	0.000704
3	0.00174	-3.73e-05

"Cylindrical Tank" (F1) with Local matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

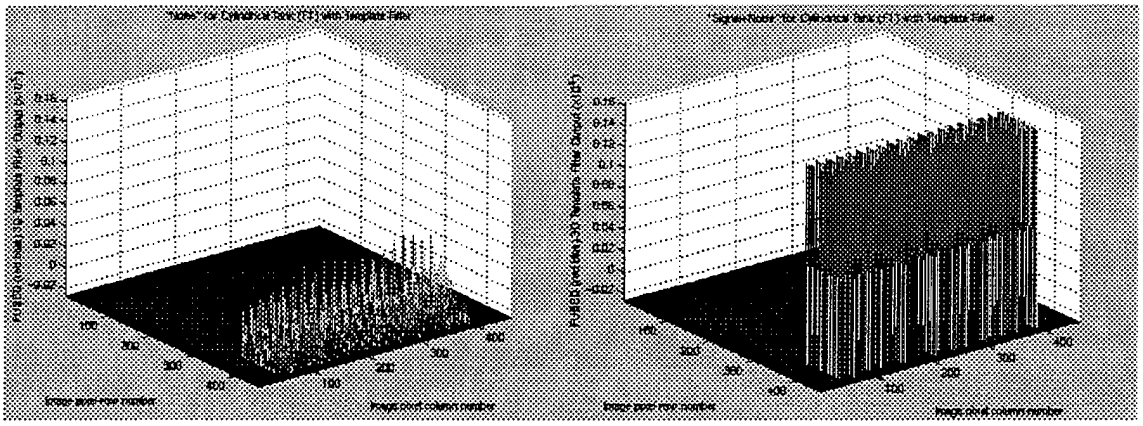


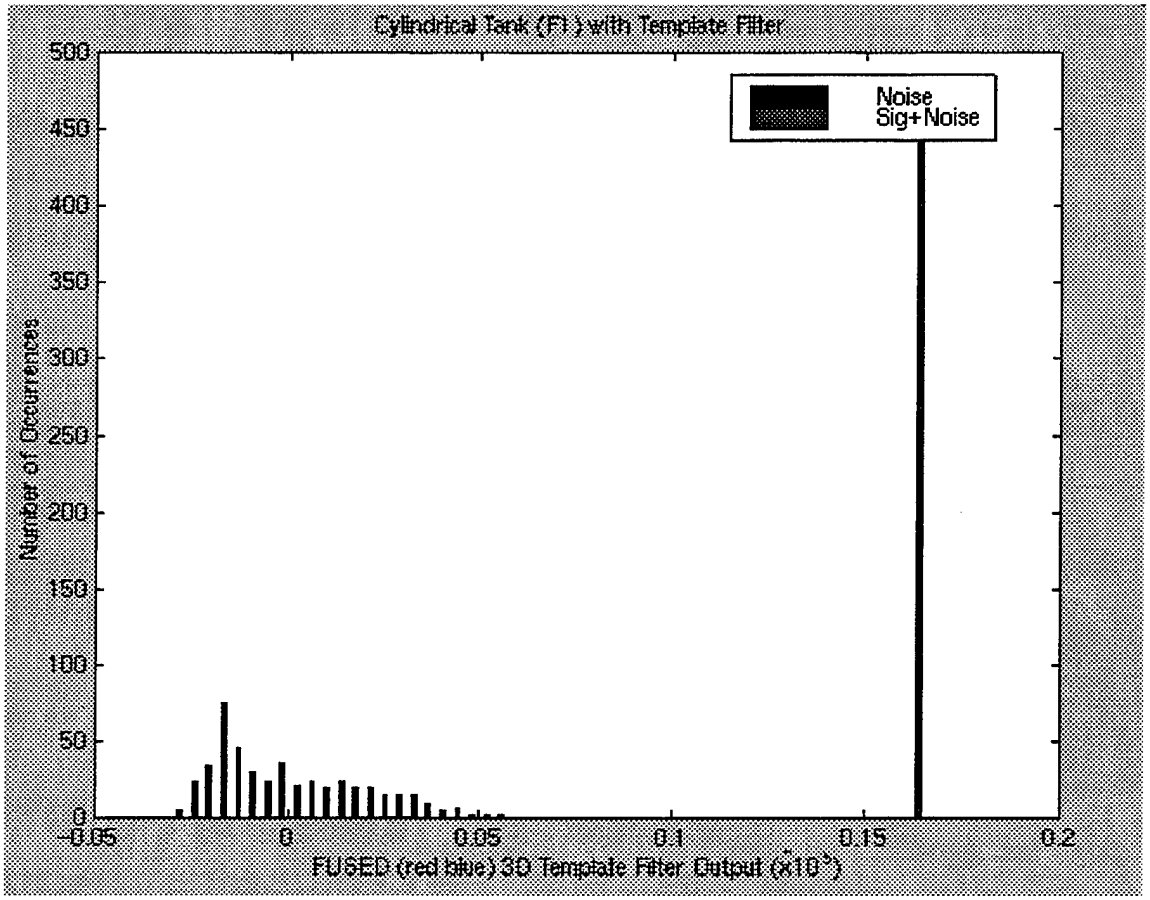


Threshold Criteria

Target Position	SNt	Nt
1	0.012	-0.00235
2	0.0102	-0.00119
3	0.0118	-0.00158

"Cylindrical Tank" (F1) with Template matching filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

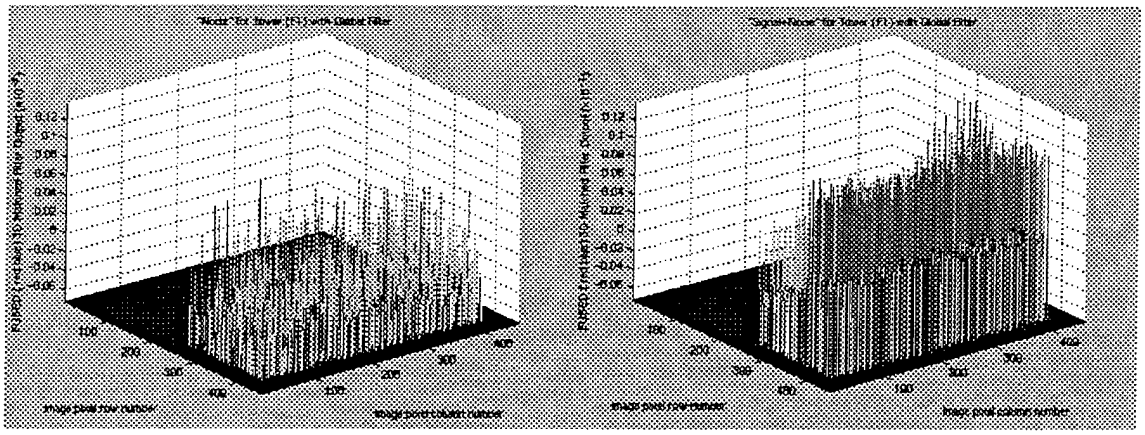


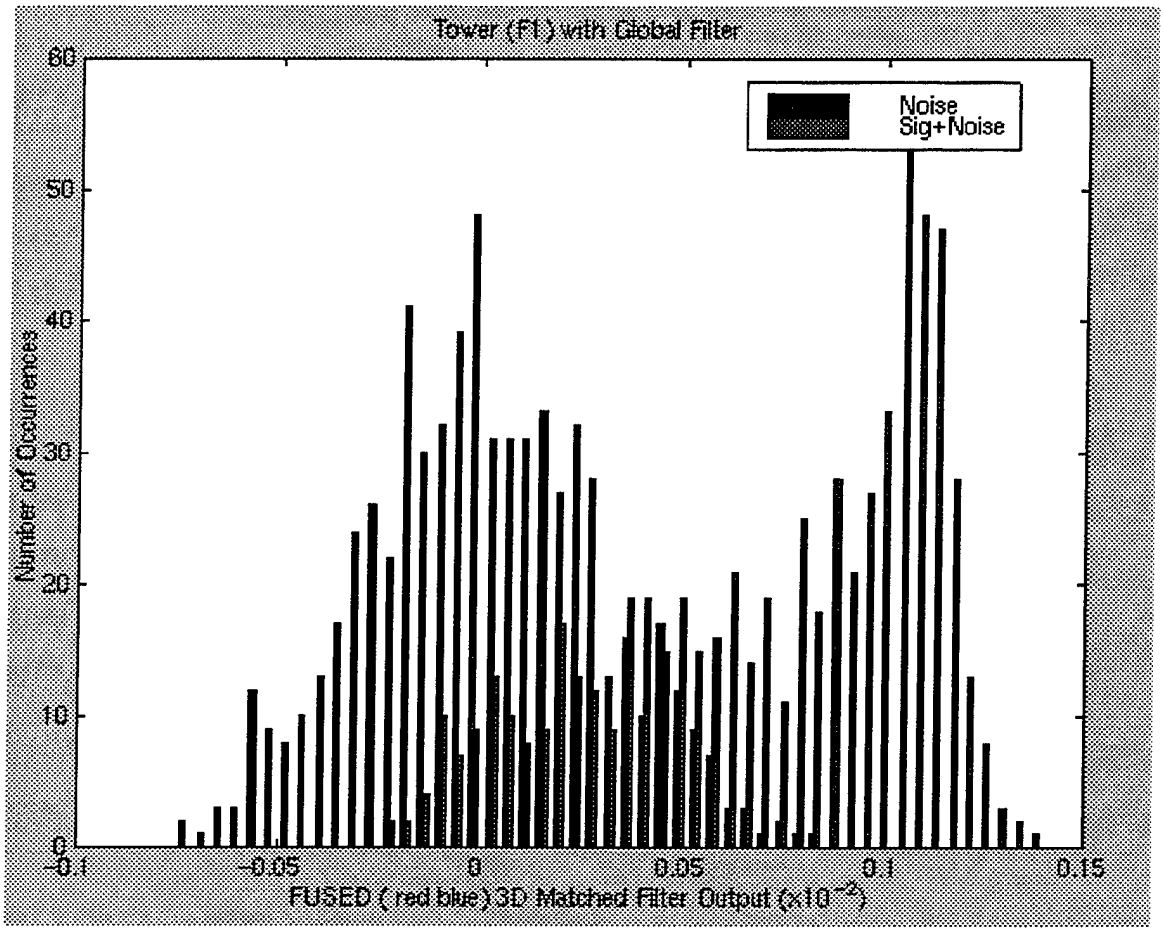


Threshold Criteria

Target Position	SNt	Nt
1	1.63e+04	3.96e+03
2	1.63e+04	3.55e+03
3	1.63e+04	1.62e+03

"Tower" (F1) with Global matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

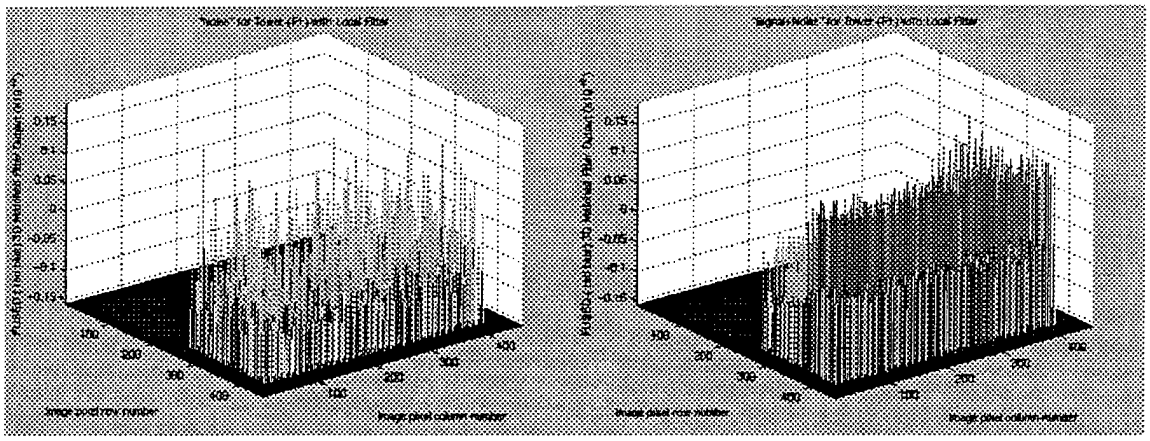


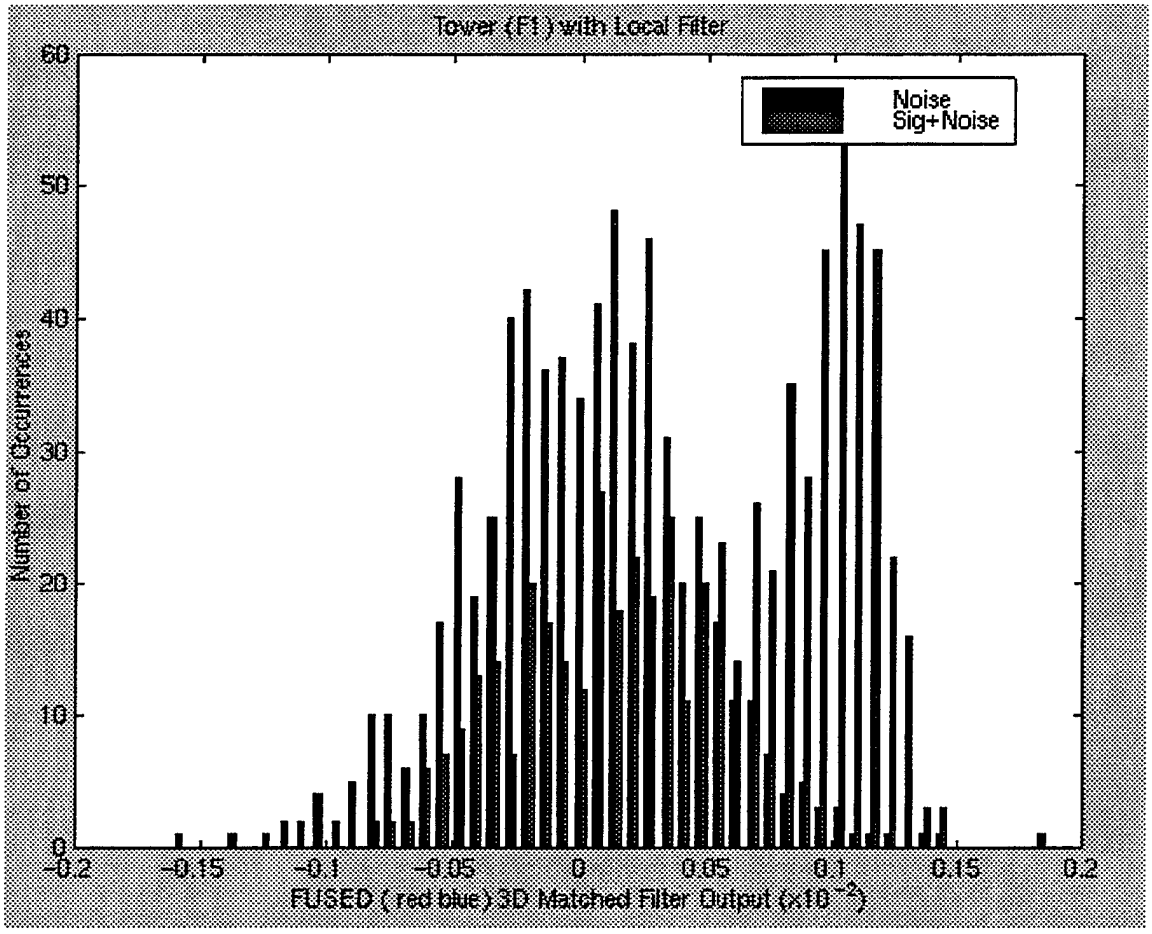


Threshold Criteria

Target Position	SNt	Nt
1	0.000763	0.000211
2	0.000926	-0.000148
3	0.000801	-4.4e-05

"Tower" (F1) with Local matched filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

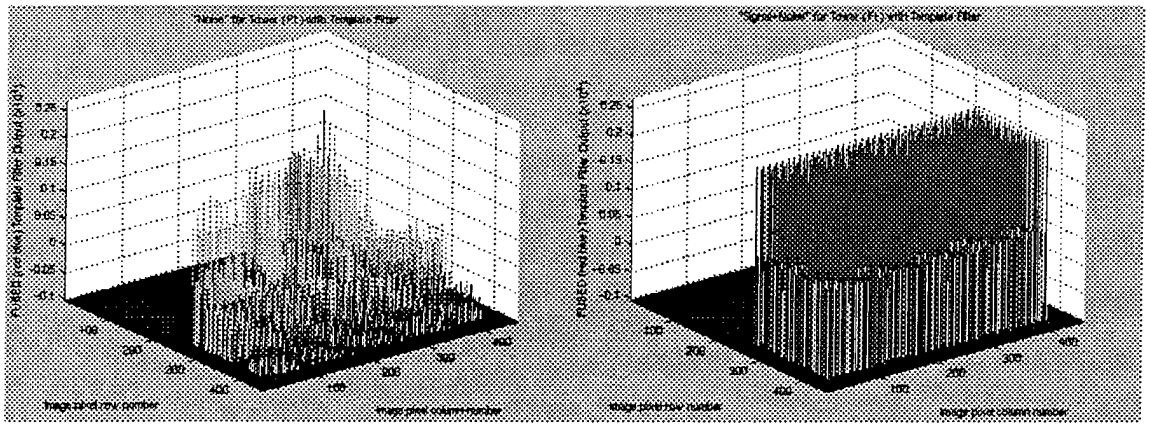


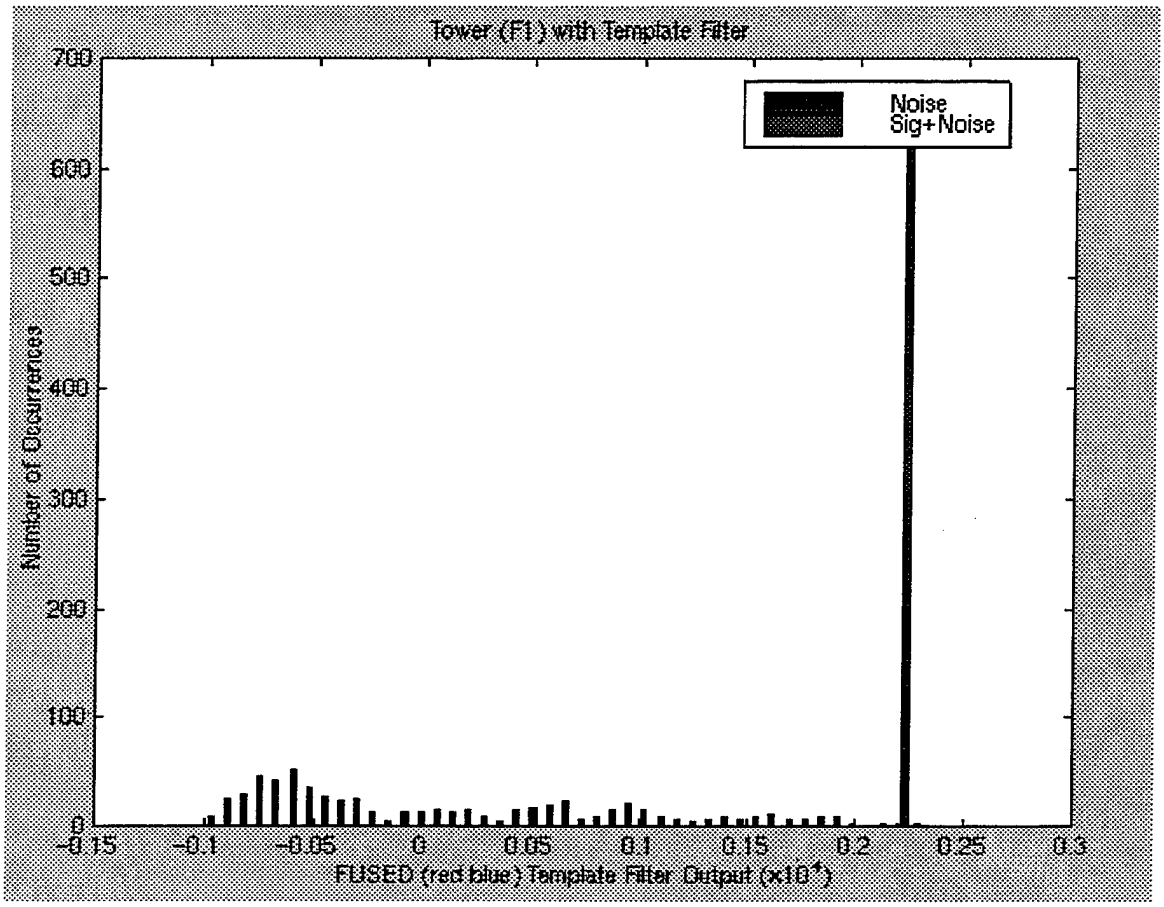


Threshold Criteria

Target Position	SNt	Nt
1	0.000529	0.000421
2	0.000886	-0.000267
3	0.000742	-0.00051

"Tower" (F1) with Template matching filter's noise data, signal+noise data, histogram, and treshold critera for each of the target positions.

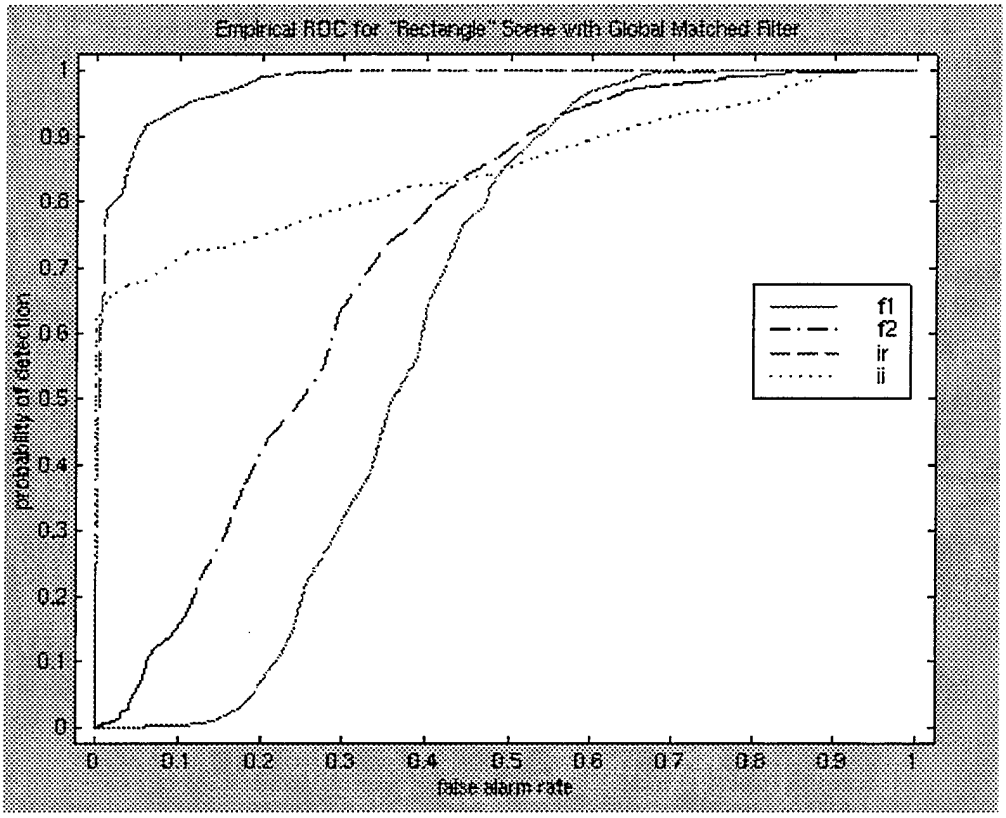


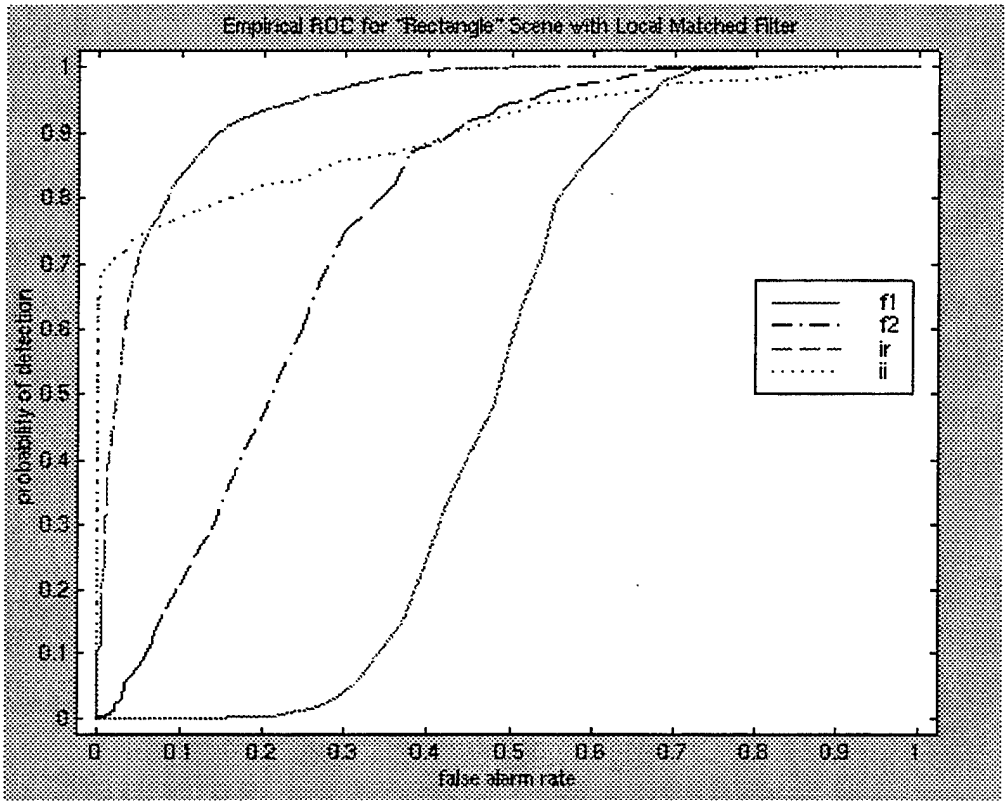


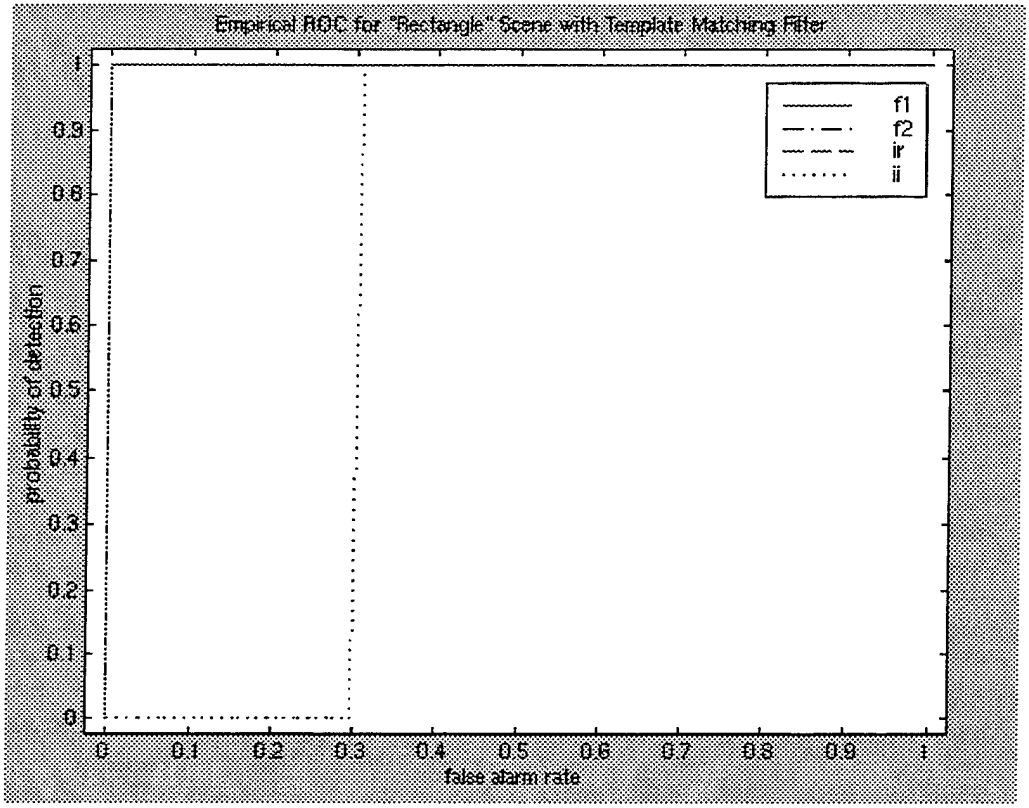
Threshold Criteria

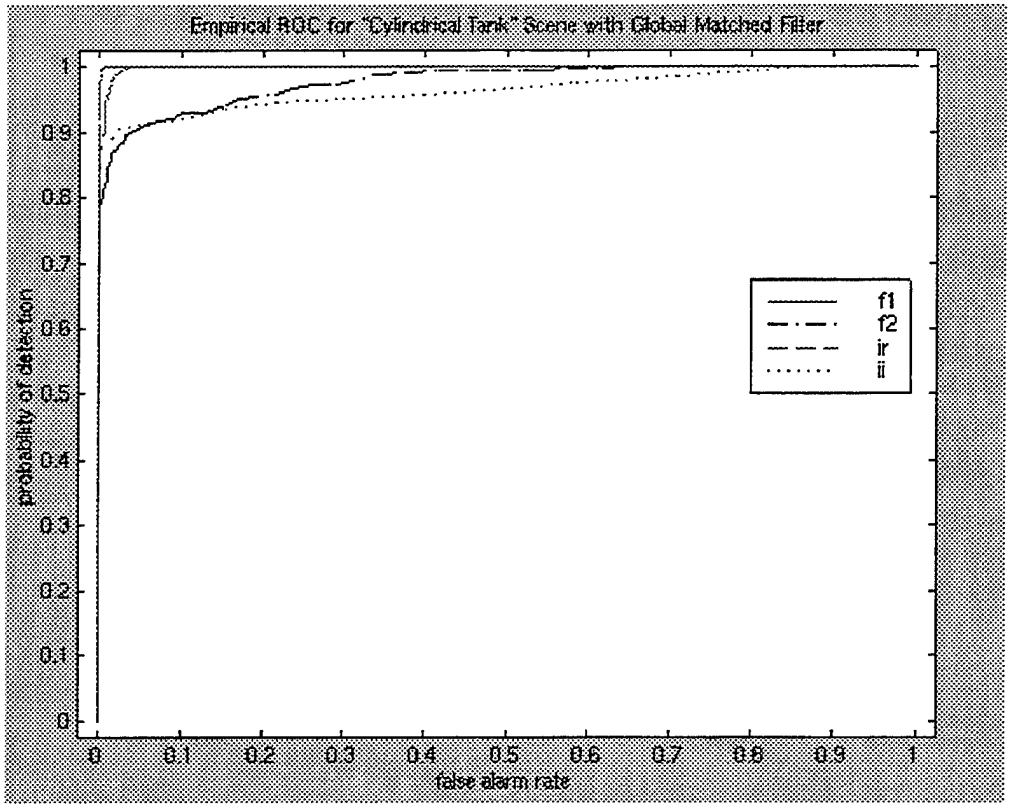
Target Position	SNt	Nt
1	2.24e+03	230
2	2.24e+03	-573
3	2.24e+03	-300

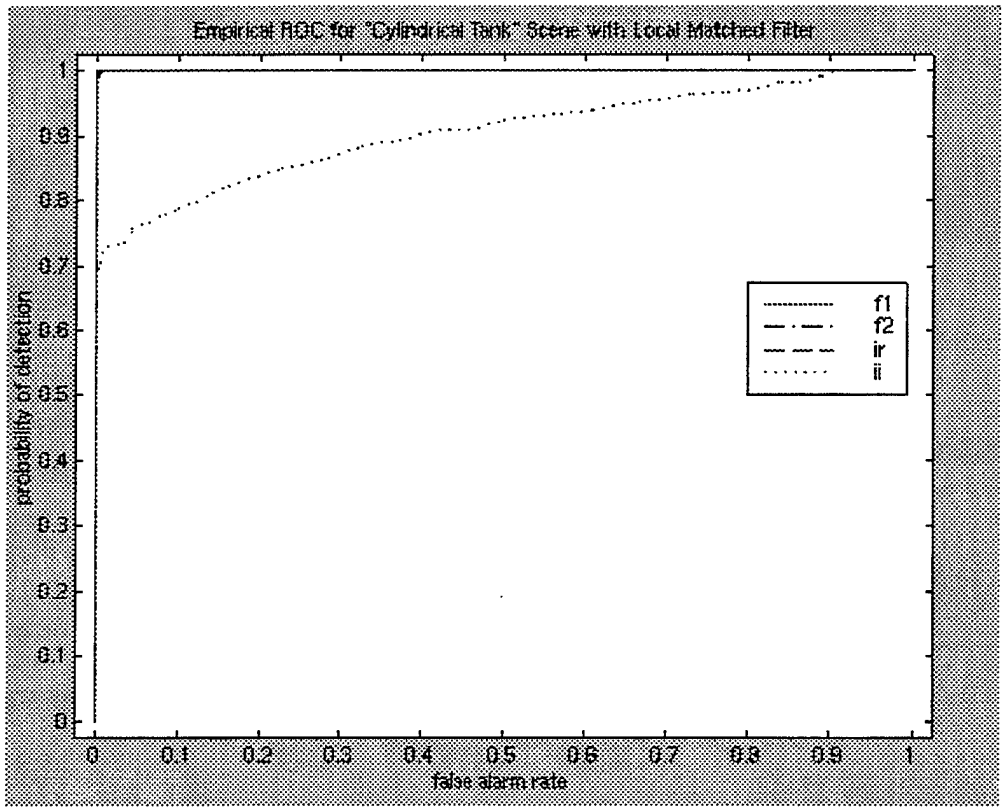
APPENDIX O. EMPRIRCALLY DERIVED ROC PLOTS

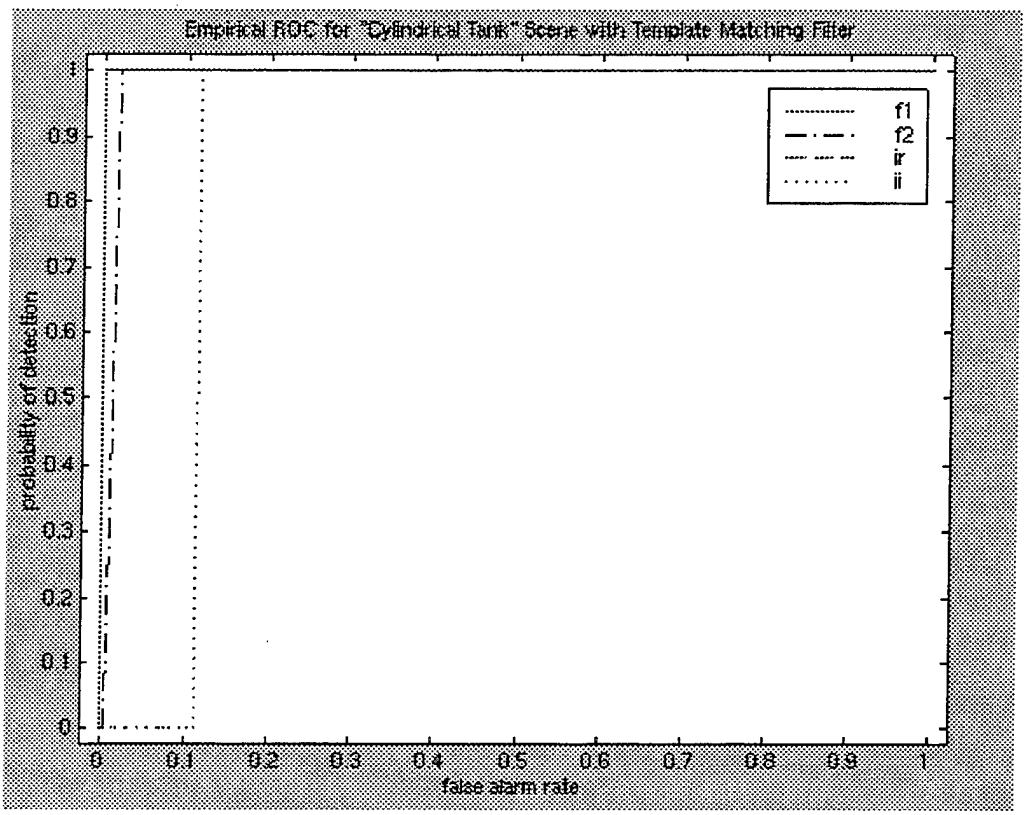


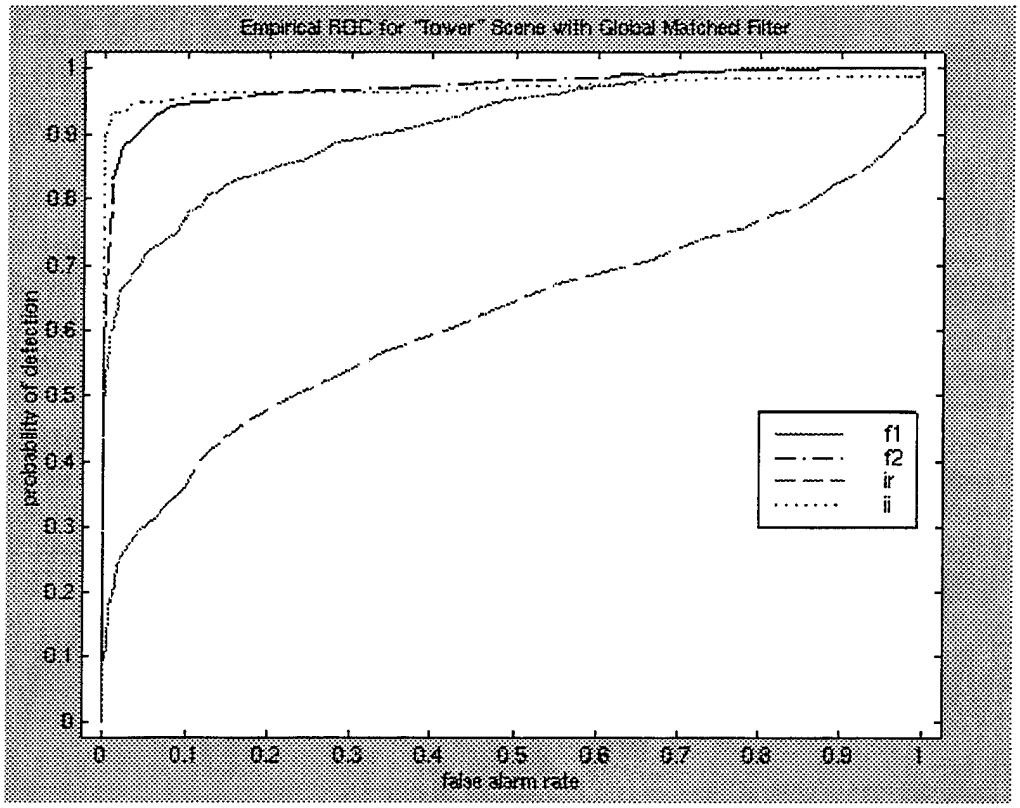


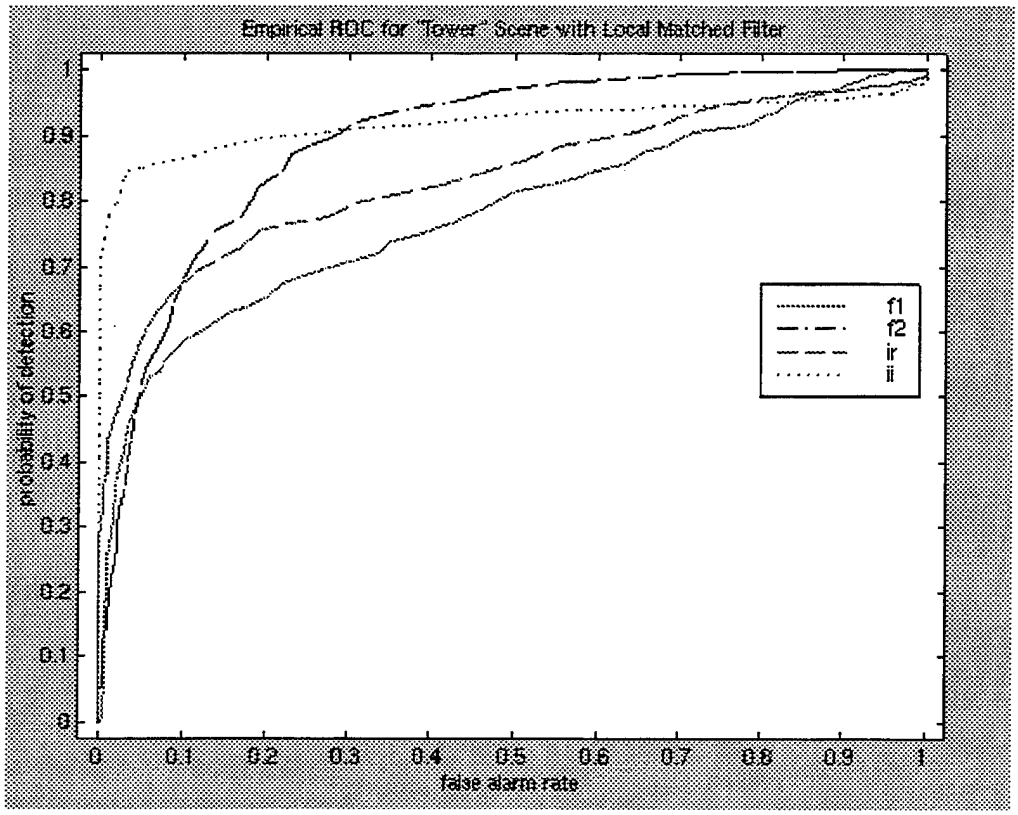

Empirical ROC for "Rectangle" Scene with Template Matching Filter
probability of detection
false alarm rate
f1
f2
ir
ii
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

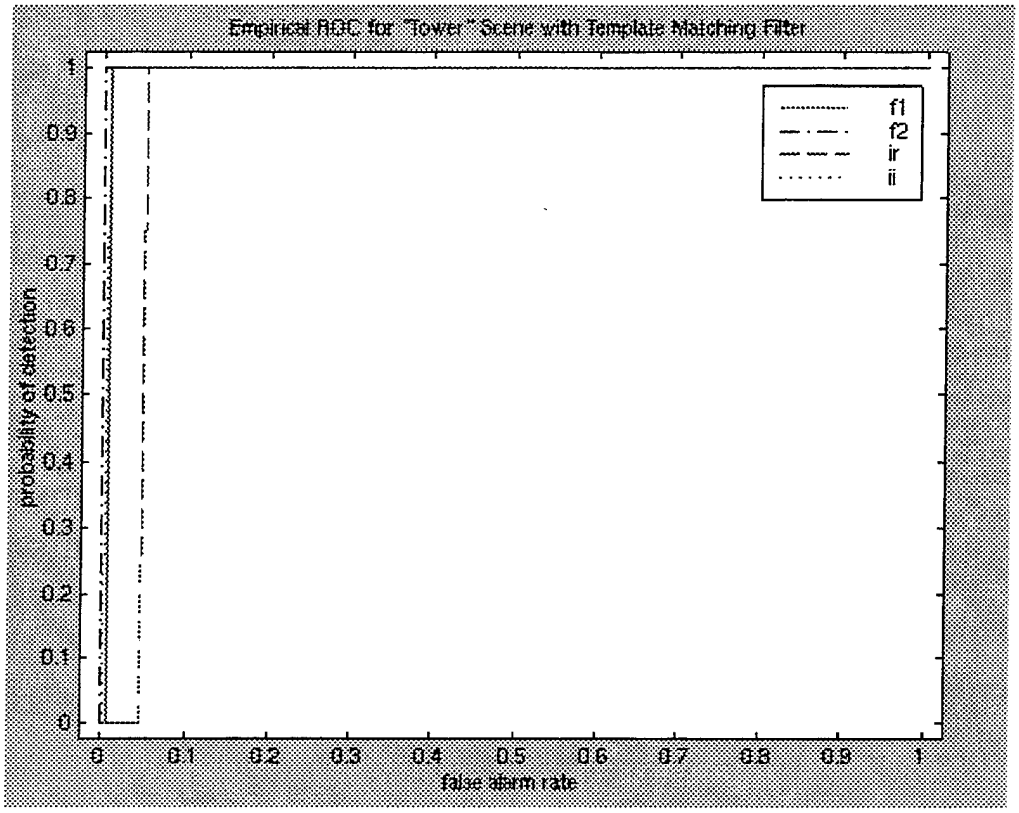

Empirical ROC for "Cylindrical Tank" Scene with Global Matched Filter
probability of detection
false alarm rate
f1
f2
ir
ii


Empirical ROC for "Cylindrical Tank" Scene with Local Matched Filter
probability of detection
false alarm rate
f1
f2
ir
ii


Empirical ROC for "Cylindrical Tank" Scene with Template Matching Filter
probability of detection
false alarm rate
f1
f2
ir
ii
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1


Empirical ROC for "Tower" Scene with Global Matched Filter
probability of detection
false alarm rate
f1
f2
ir
ii


Empirical ROC for "Tower" Scene with Local Matched Filter
probability of detection
false alarm rate
f1
f2
ir
ii

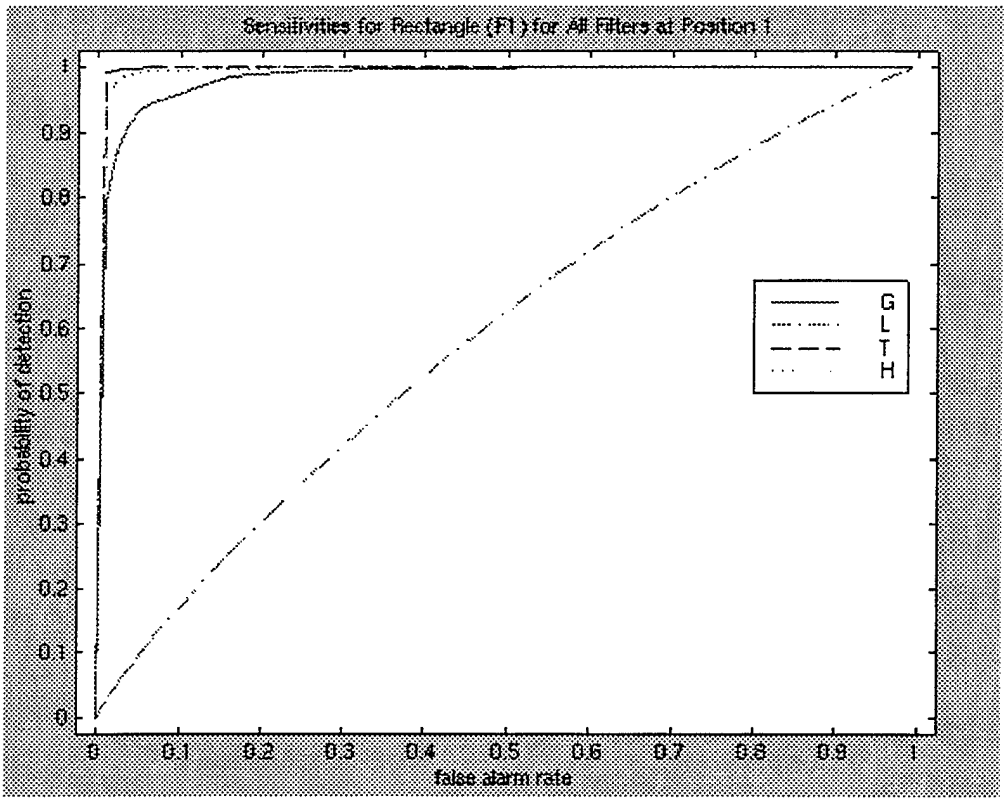

Empirical ROC for "Tower" Scene with Template Matching Filter
f1
f2
ir
ii
probability of detection
false alarm rate
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

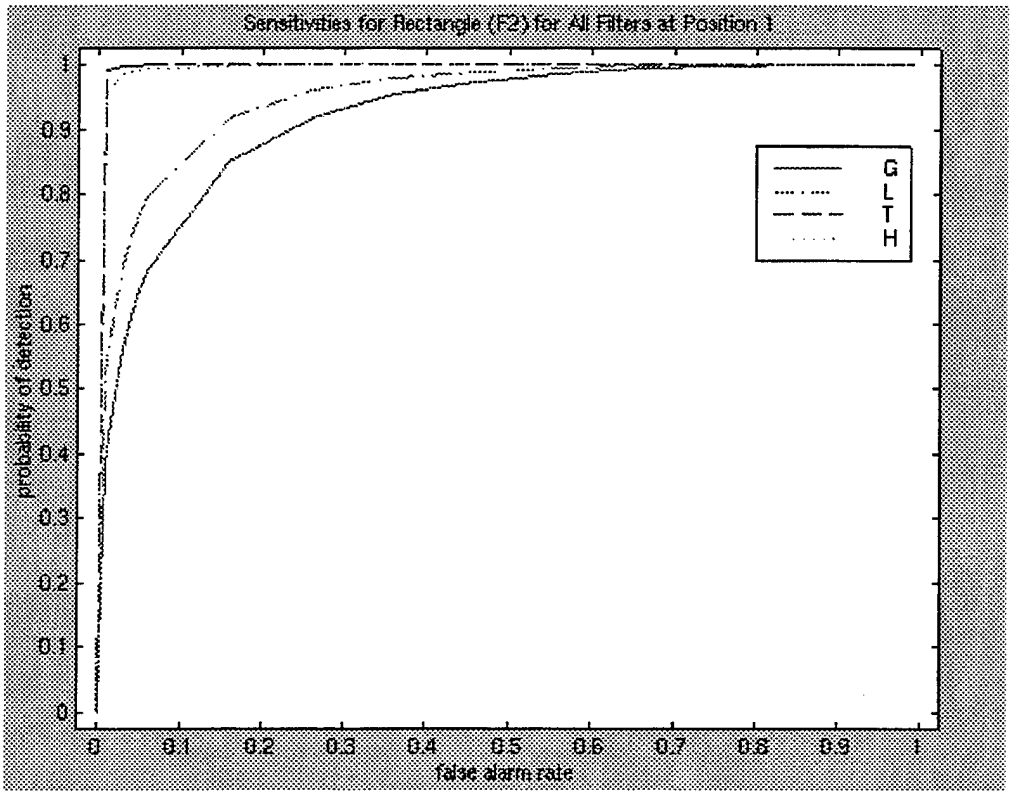
APPENDIX P. RESPONSE BIAS SUMMARY FOR EACH FILTER

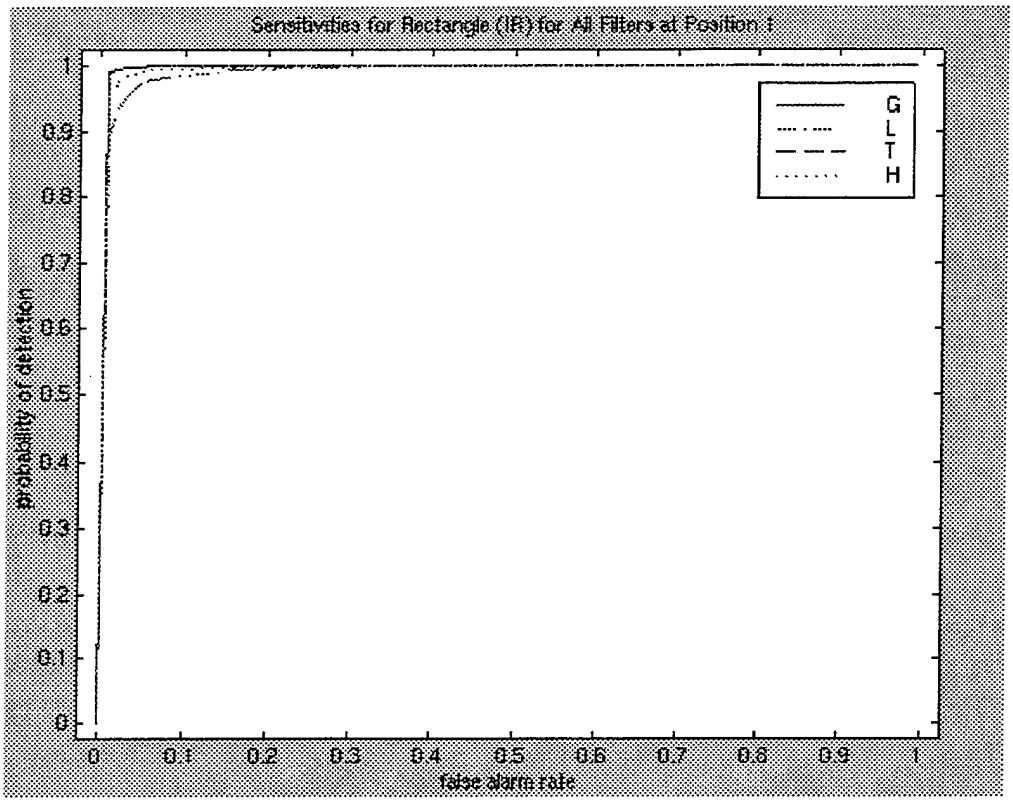
Sensor	Scene	Position	Global	Local	Template	Human
F1	rct	1	0.626	0.9259	0.6126	0.62016
F2	rct	1	0.677	0.6552	0.6126	0.62016
IR	rct	1	0.6126	0.6172	0.6126	0.62016
IL	rct	1	0.7138	0.6648	0.7215	0.621386
F1	tnk	1	0.6126	0.6126	0.6126	0.612626
F2	tnk	1	0.6168	0.6126	0.6127	0.612747
IR	tnk	1	0.6126	0.6126	0.6126	0.612747
IL	tnk	1	0.6185	0.8824	0.6726	0.613201
F1	twr	1	0.682	0.7378	0.6126	0.620038
F2	twr	1	0.6129	0.6667	0.6126	0.620038
IR	twr	1	0.7604	0.7337	0.6133	0.739722
IL	twr	1	0.6281	0.7176	0.6126	0.620038
F1	rct	2	0.7638	0.6676	0.6126	0.620038
F2	rct	2	0.7048	0.6392	0.6126	0.686942
IR	rct	2	0.6126	0.6126	0.6126	0.620038
IL	rct	2	0.826	0.7031	0.7029	0.673672
F1	tnk	2	0.6126	0.6126	0.6126	0.612626
F2	tnk	2	0.6126	0.6126	0.6127	0.612626
IR	tnk	2	0.6126	0.6126	0.6126	0.612626
IL	tnk	2	0.6321	0.8941	0.6541	0.612626
F1	twr	2	0.6126	0.6608	0.6126	0.620038
F2	twr	2	0.6234	0.615	0.6126	0.620038
IR	twr	2	0.8301	0.6904	0.6133	0.612626
IL	twr	2	0.6281	0.6542	0.6126	0.612626
F1	rct	3	0.7229	0.6427	0.6126	0.612626
F2	rct	3	0.6769	0.6514	0.6126	0.62016
IR	rct	3	0.6126	0.6126	0.6126	0.612626
IL	rct	3	0.7536	0.6835	0.7153	0.633501
F1	tnk	3	0.6126	0.6126	0.6126	0.612626
F2	tnk	3	0.6222	0.6126	0.6127	0.612626
IR	tnk	3	0.6126	0.6126	0.6126	0.620038
IL	tnk	3	0.7251	0.9257	0.6316	0.620038
F1	twr	3	0.6271	0.6258	0.6126	0.612626
F2	twr	3	0.6169	0.6639	0.6126	0.612626
IR	twr	3	0.8298	0.6283	0.6133	0.620038
IL	twr	3	0.6365	0.644	0.6126	0.612626

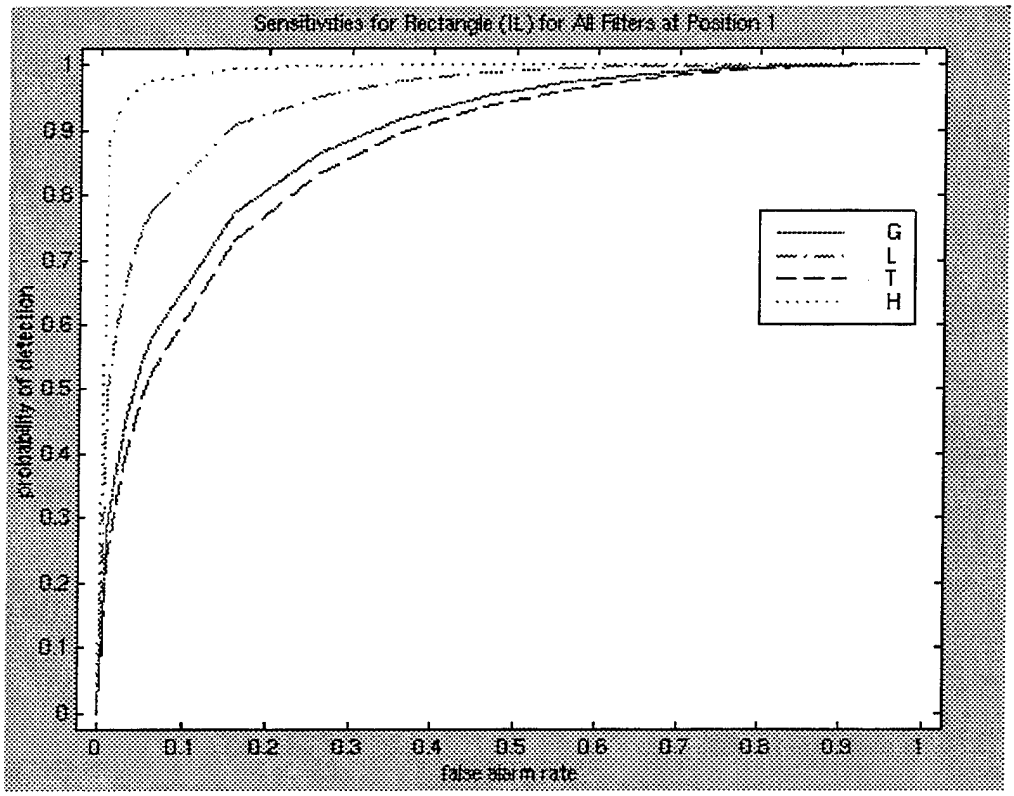
Table P1. This table summarizes the response biases for the global and local matched filters, the response biases for the template matching filter, and the averaged results of the subjects participating in the human factors experiment (Sampson, 1996). For example, the last line in this table shows the response biases found for the IL tower scene with the target placed at position three.

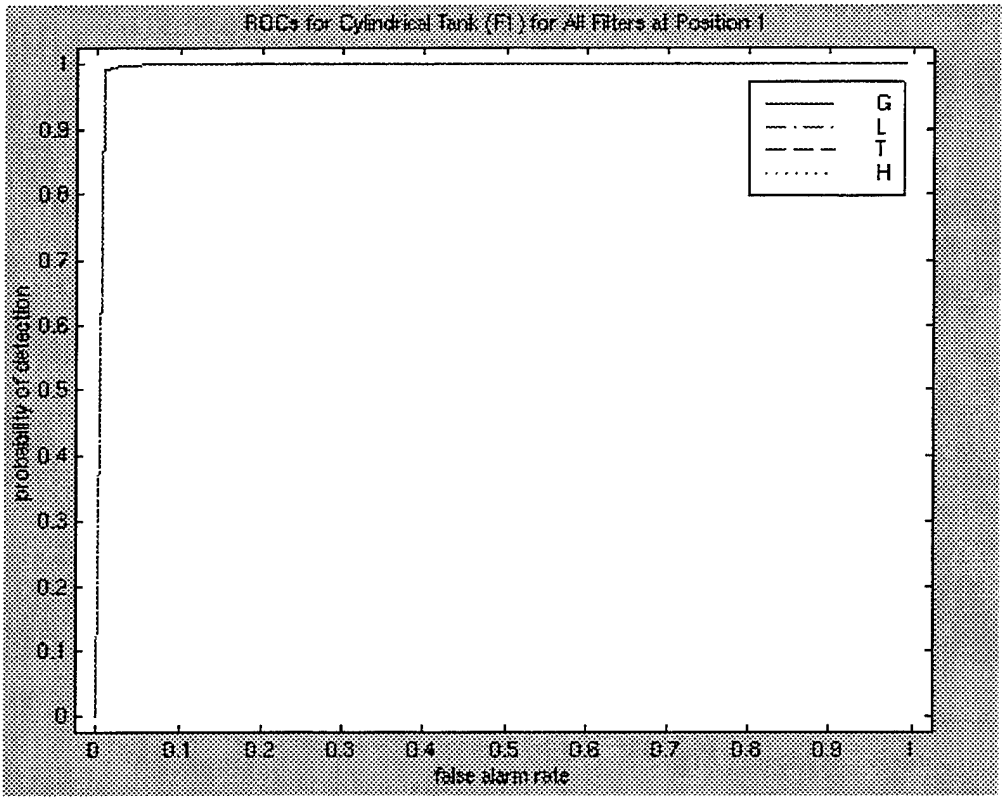
APPENDIX Q. ROC PLOTS OF SENSITIVITY RESULTS

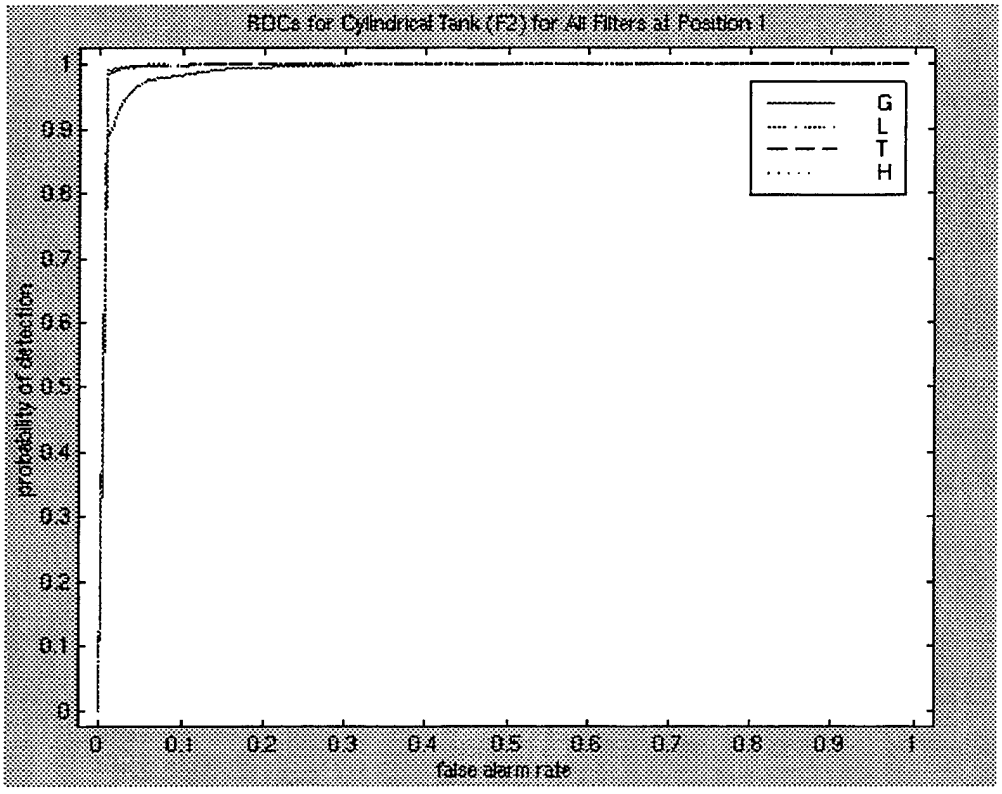


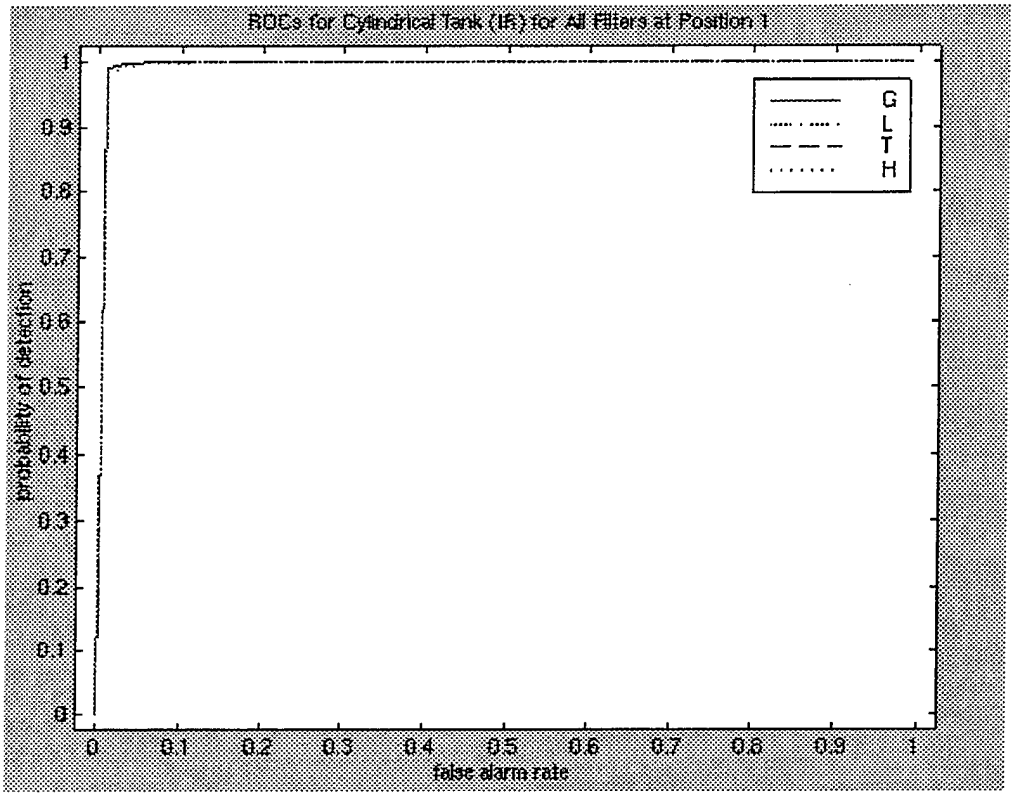


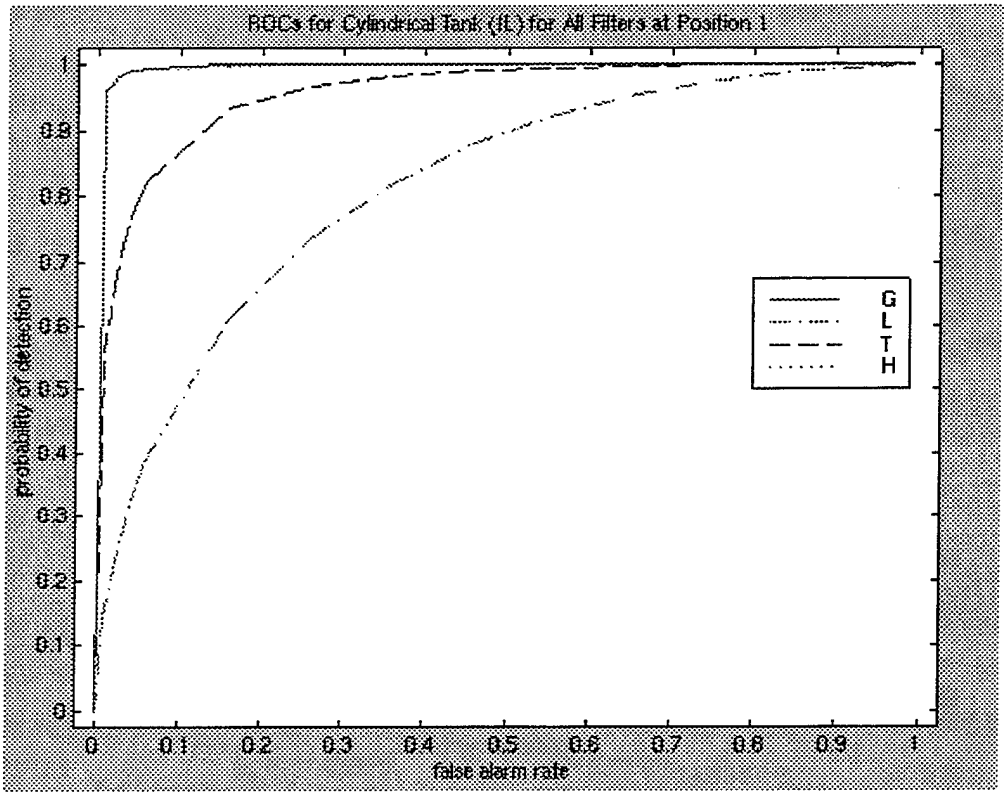

Sensitivities for Rectangle (IF) for All Filters at Position 1
probability of detection
false alarm rate
G
L
T
H

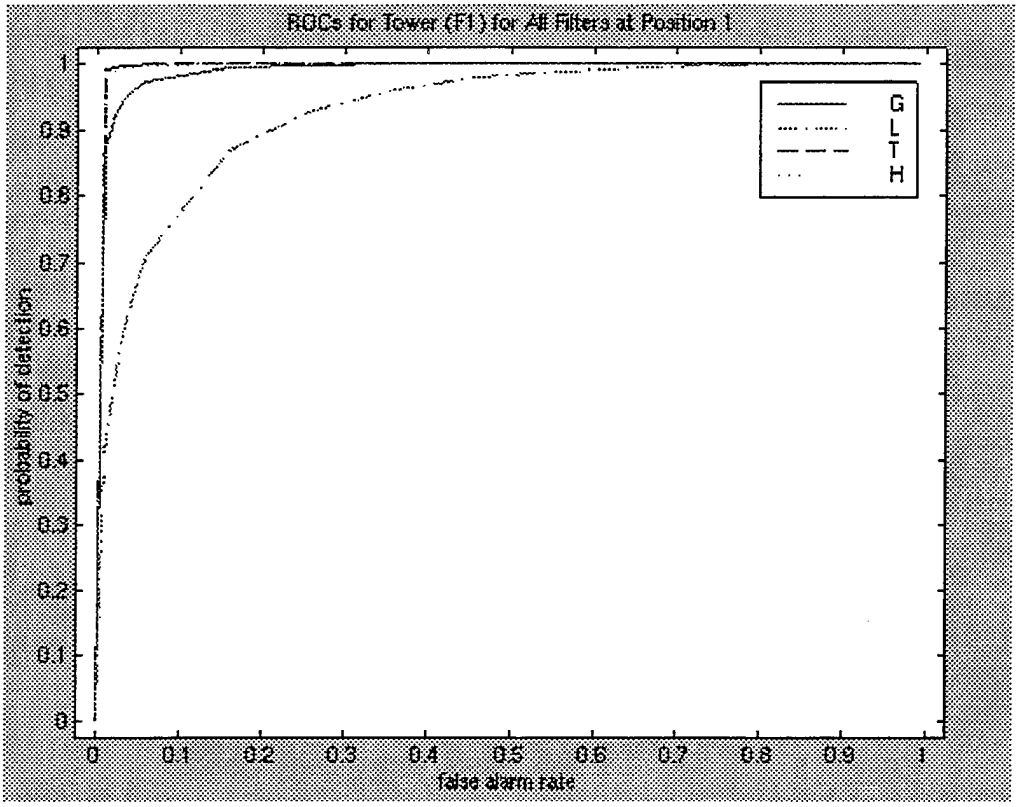

Sensitivities for Rectangle (It.) for All Filters at Position 1
probability of detection
false alarm rate
G
L
T
H

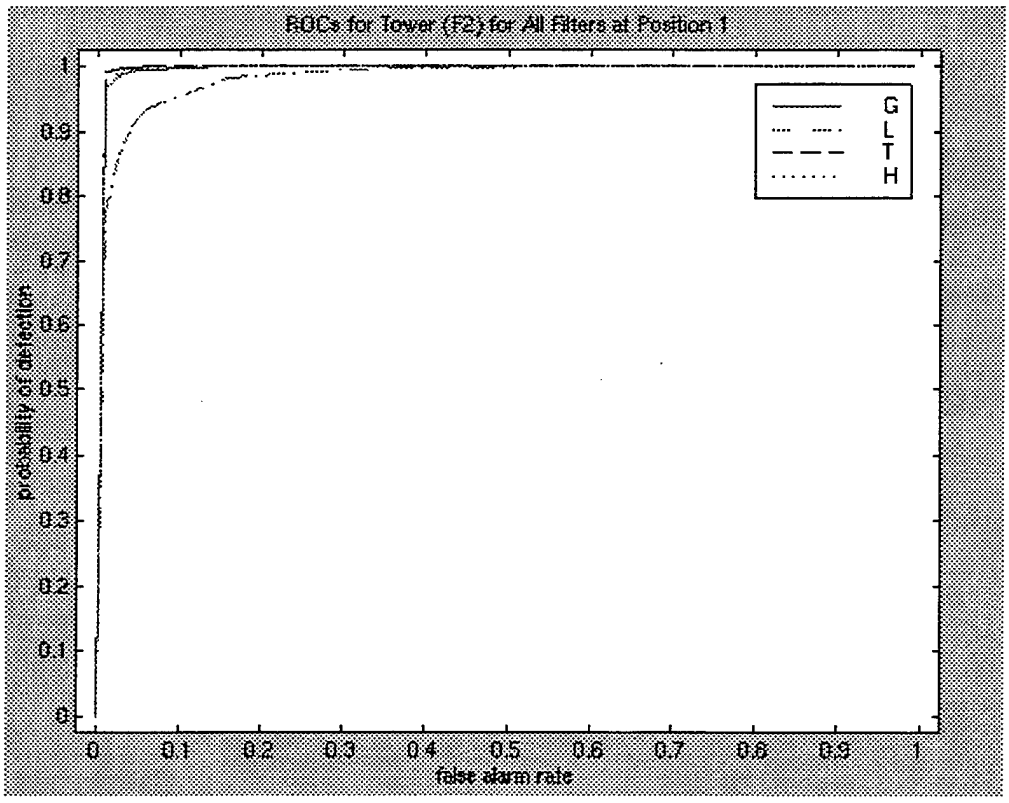

ROCs for Cylindrical Tank (F1) for All Filters at Position 1
probability of detection
false alarm rate
G
L
T
H

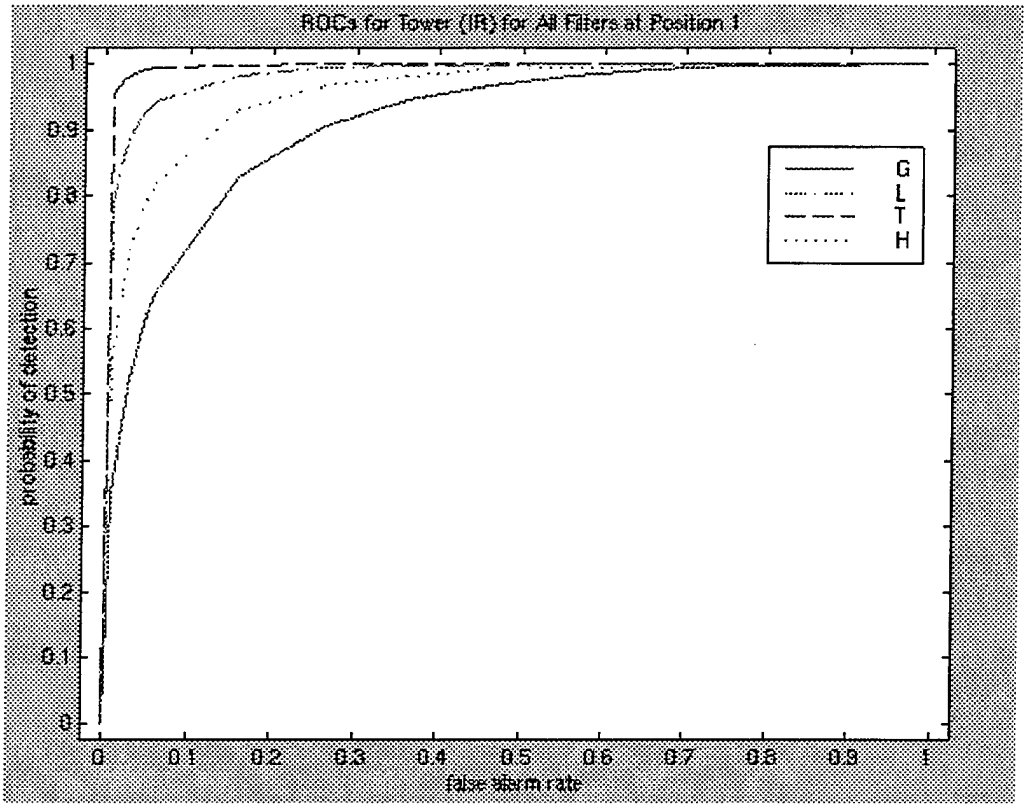

ROCs for Cylindrical Tank (F2) for All Filters at Position 1
probability of detection
false alarm rate
G
L
T
H

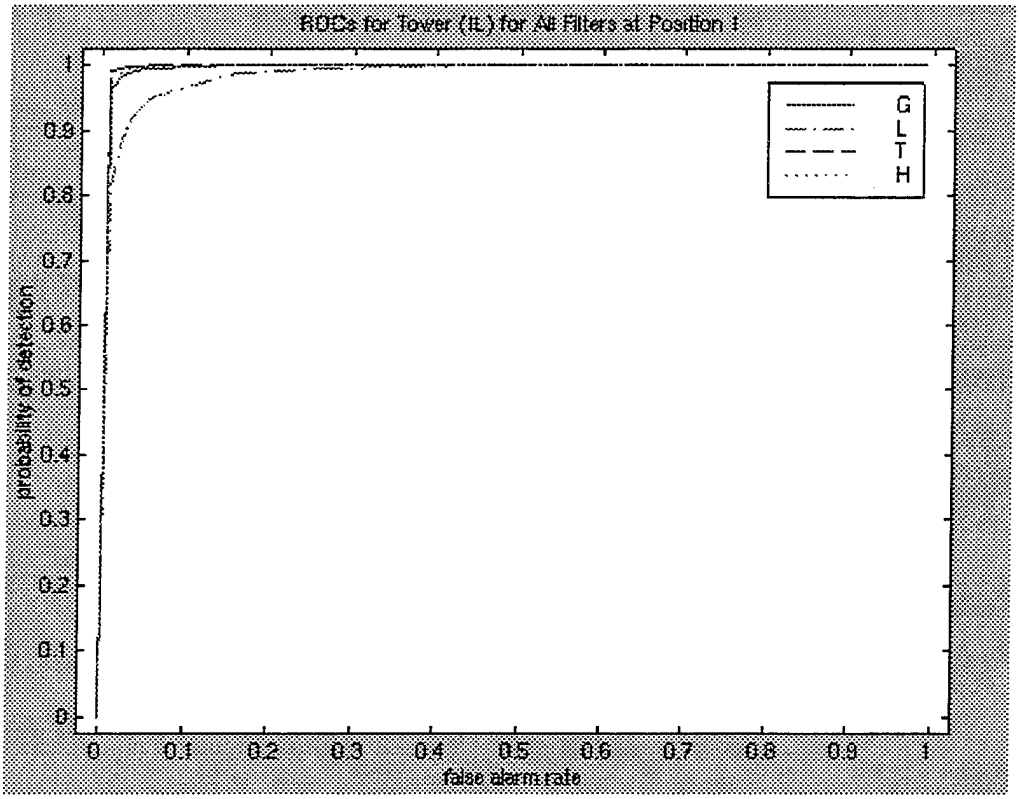

ROCs for Cylindrical Tank (IR) for All Filters at Position 1
probability of detection
false alarm rate
G
L
T
H

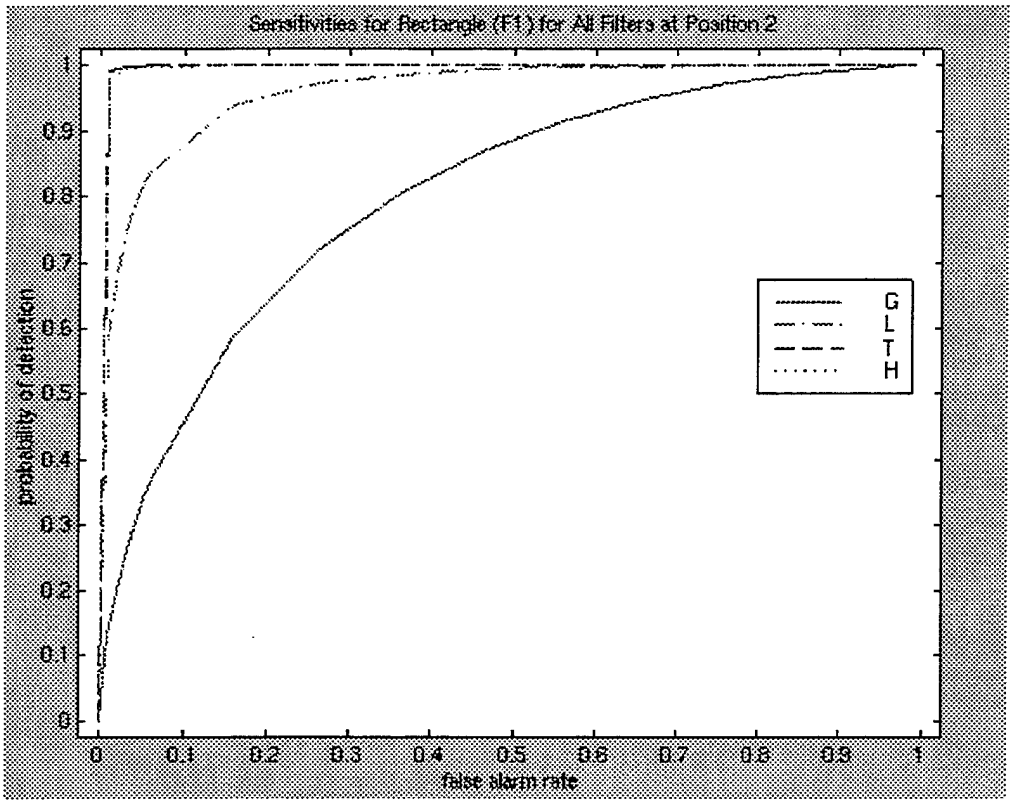

ROCs for Cylindrical Tank (IL) for All Filters at Position 1
probability of detection
false alarm rate
G
L
T
H

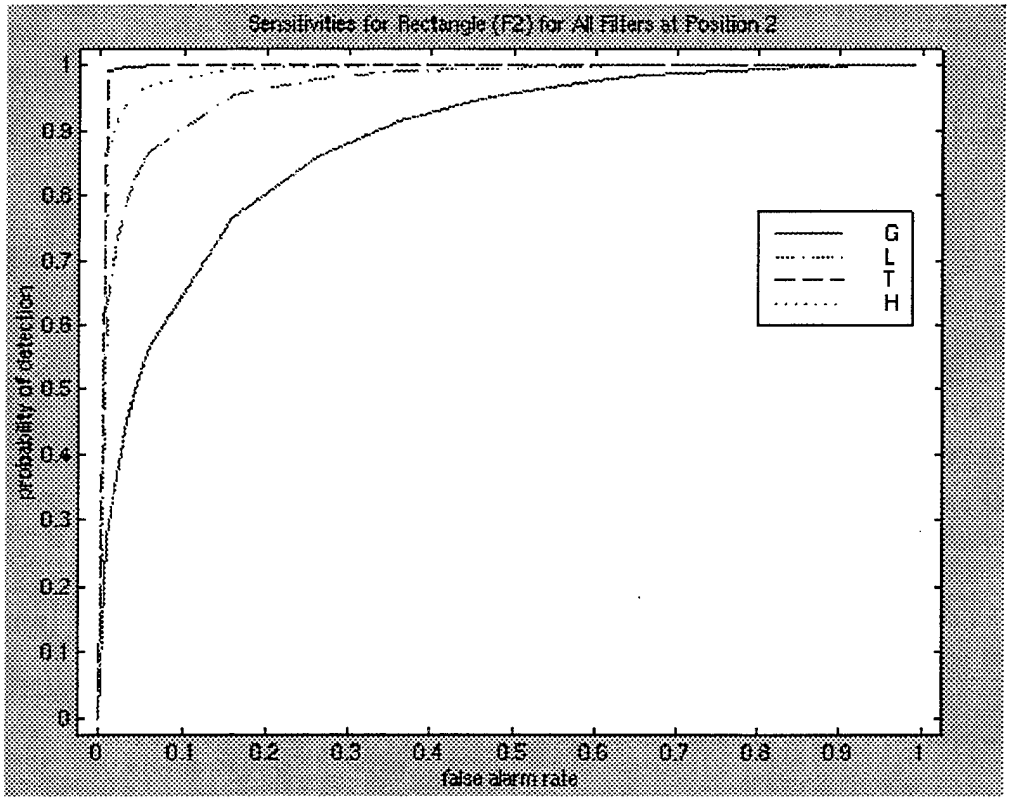

ROCs for Tower (F1) for All Filters at Position 1
probability of detection
false alarm rate
G
L
T
H

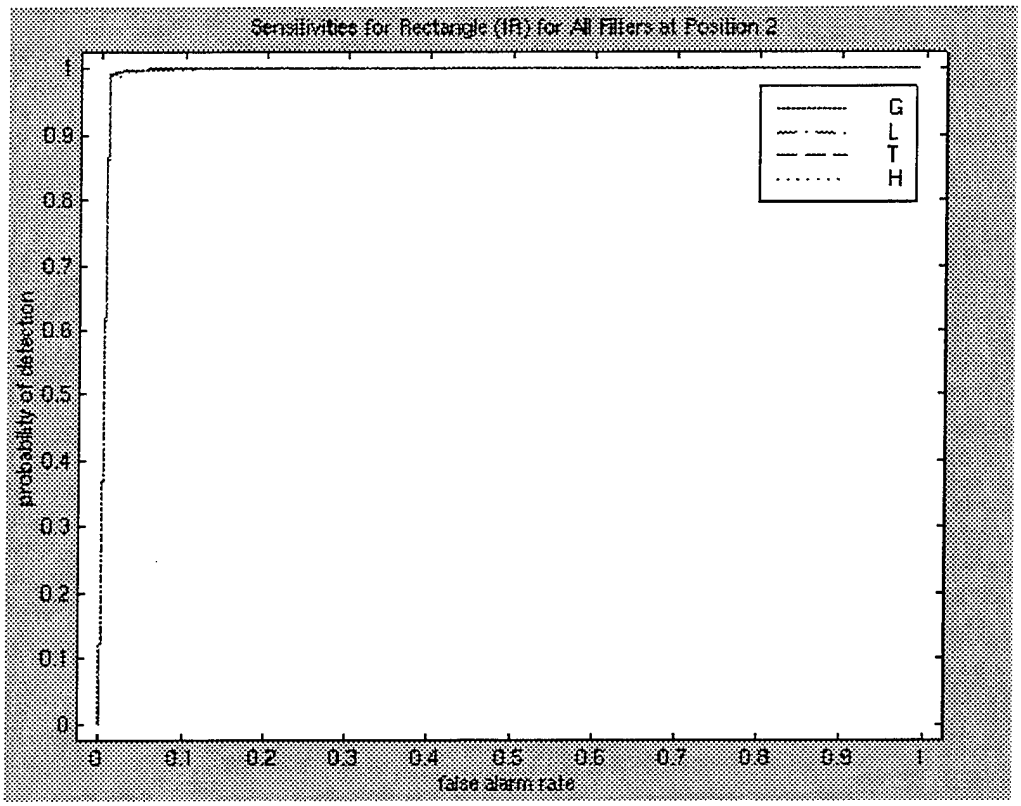

ROCs for Tower (F2) for All Filters at Position 1
probability of detection
false alarm rate
G
L
T
H

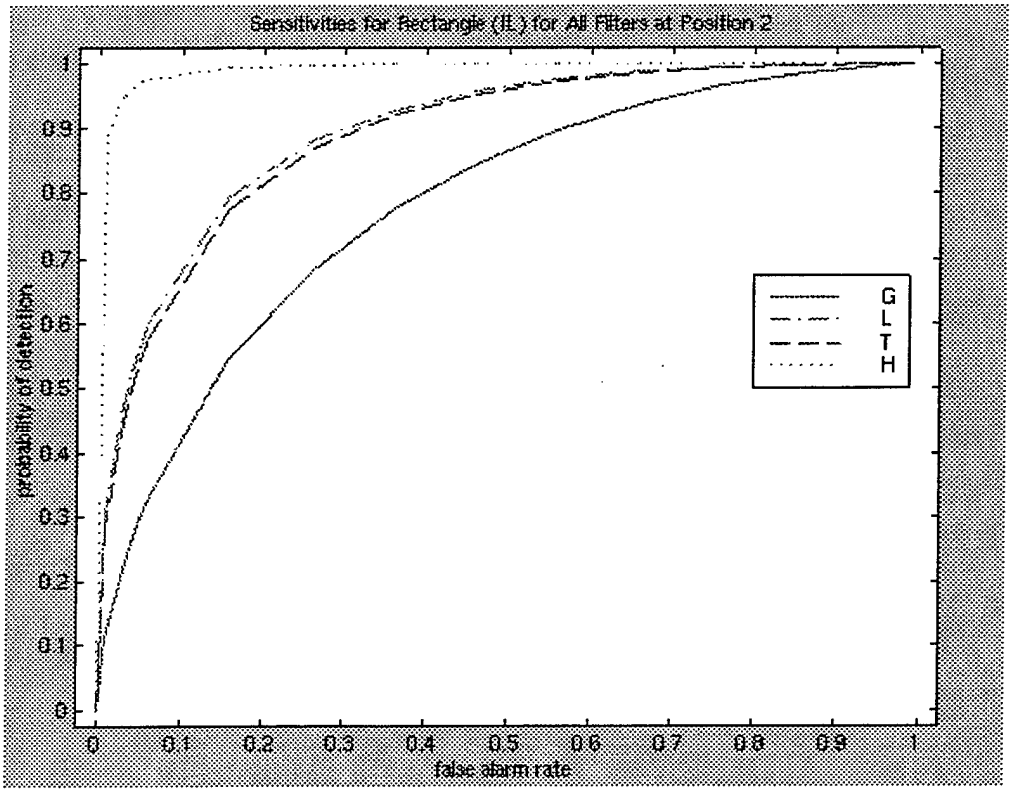

ROCs for Tower (IR) for All Filters at Position 1
probability of detection
false alarm rate
G
L
T
H

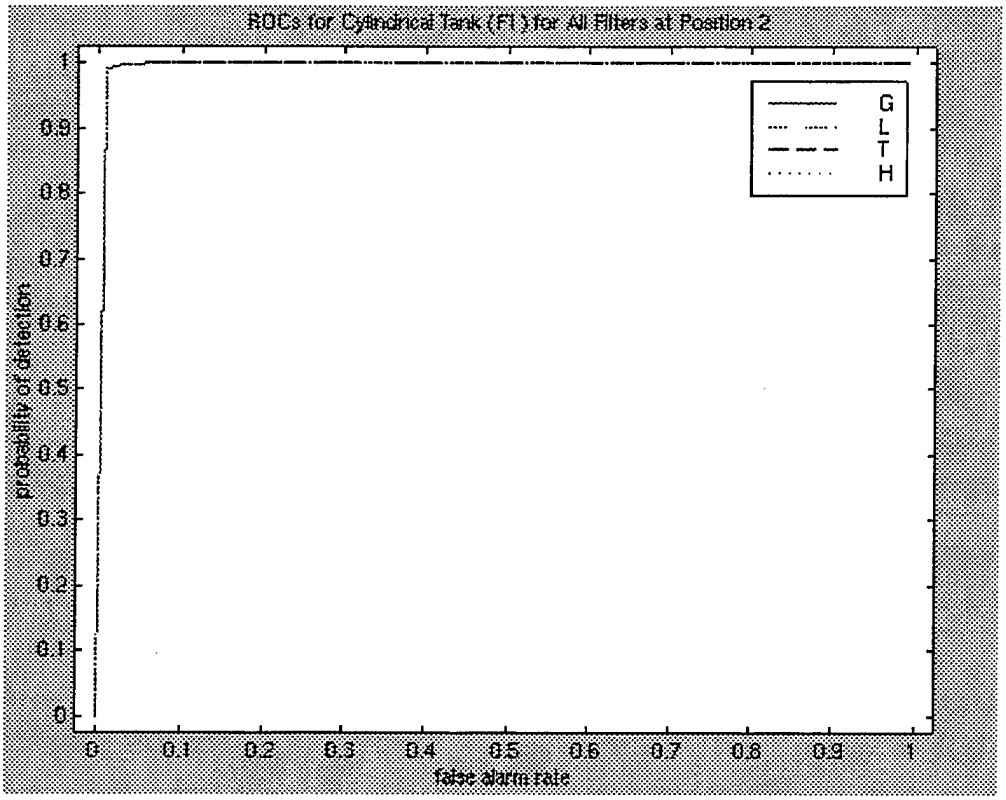

ROCs for Tower (IL) for All Filters at Position 1
probability of detection
false alarm rate
G
L
T
H

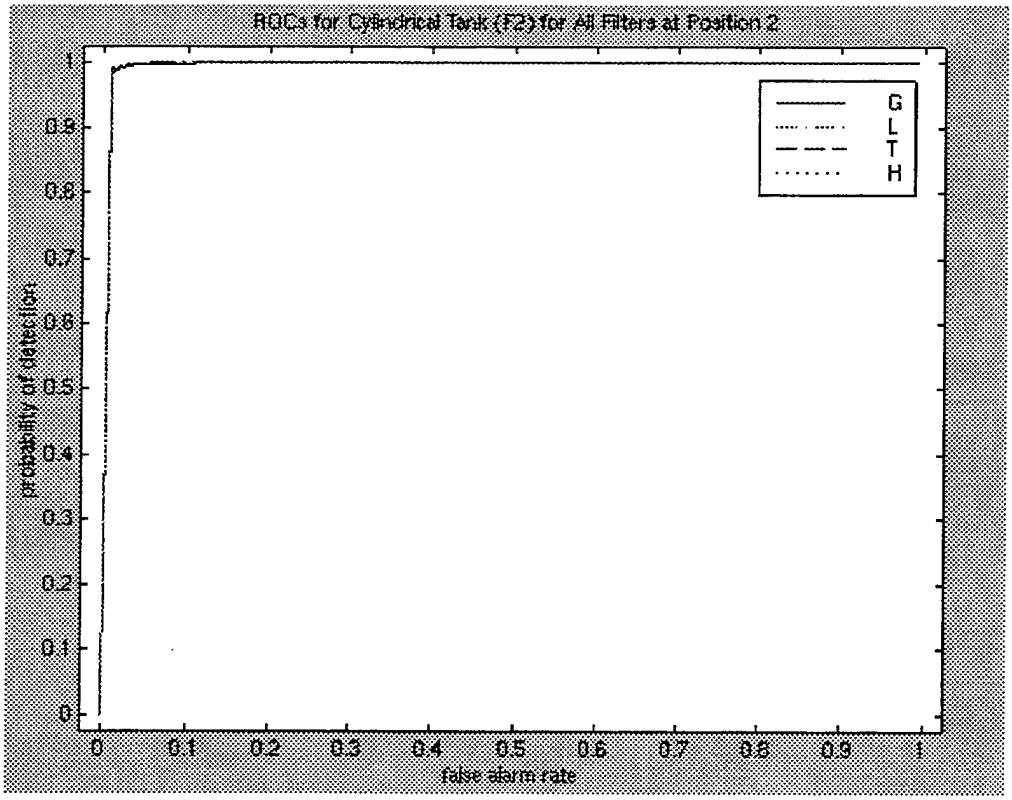

Sensitivities for Rectangle (F1) for All Filters at Position 2
probability of detection
false alarm rate
G
L
T
H

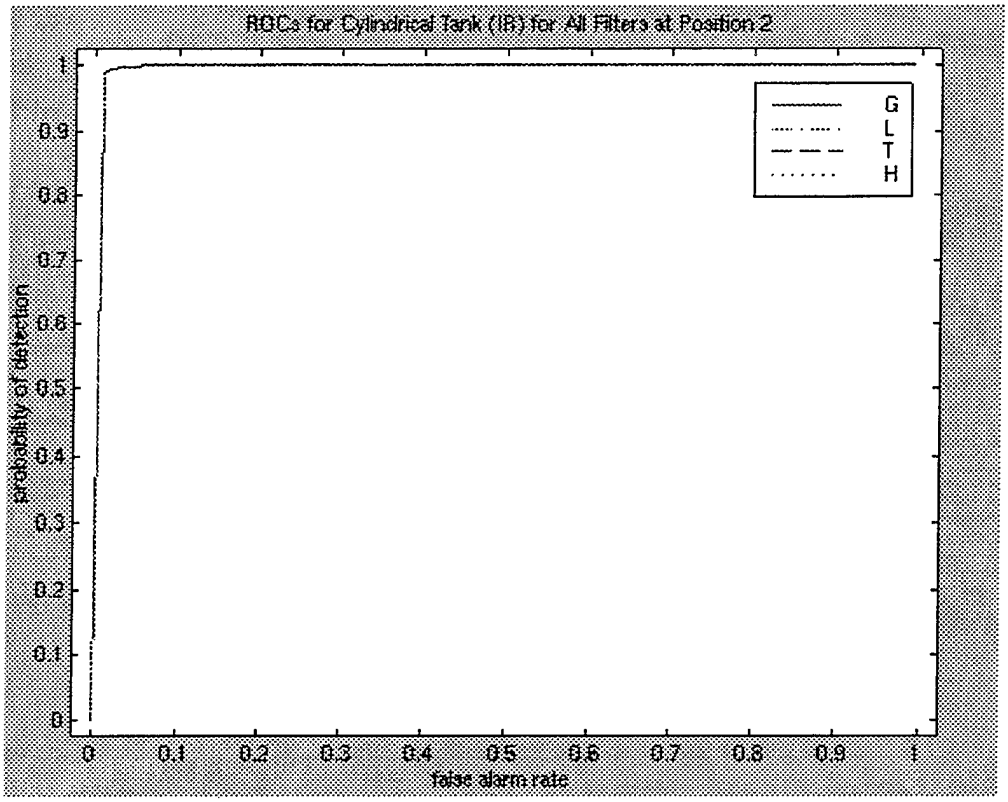

Sensitivities for Rectangle (F2) for All Filters at Position 2
probability of detection
false alarm rate
G
L
T
H

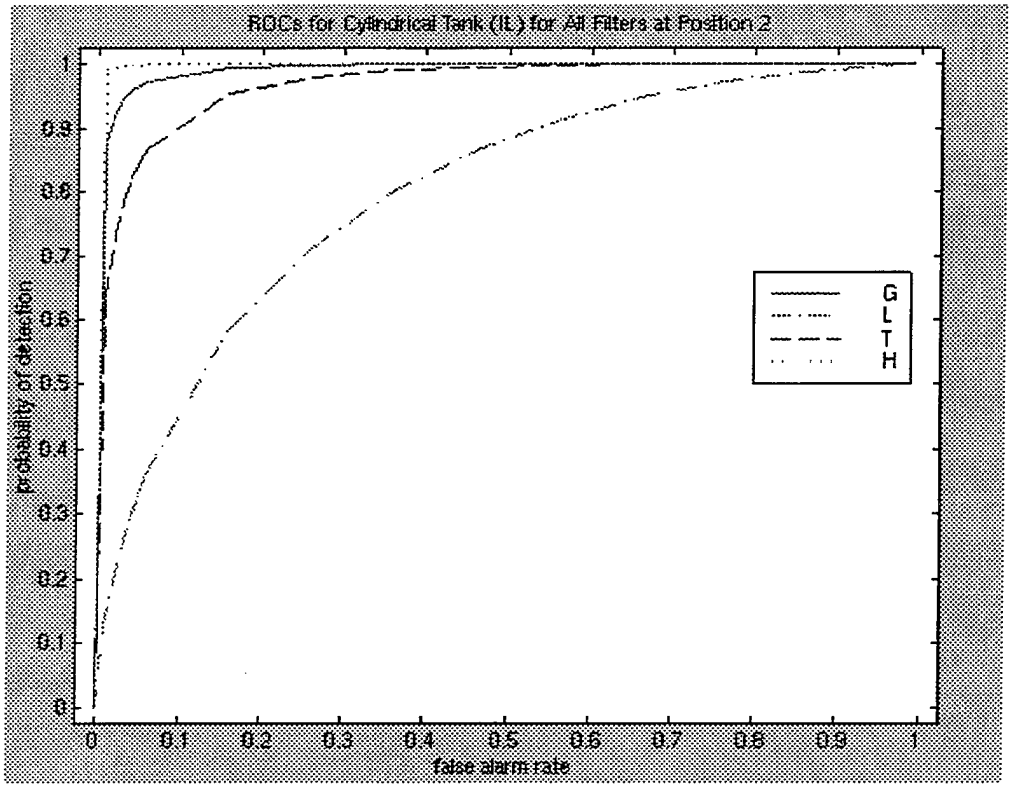

Sensitivities for Rectangle (IR) for All Filters at Position 2
probability of detection
false alarm rate
G
L
T
H

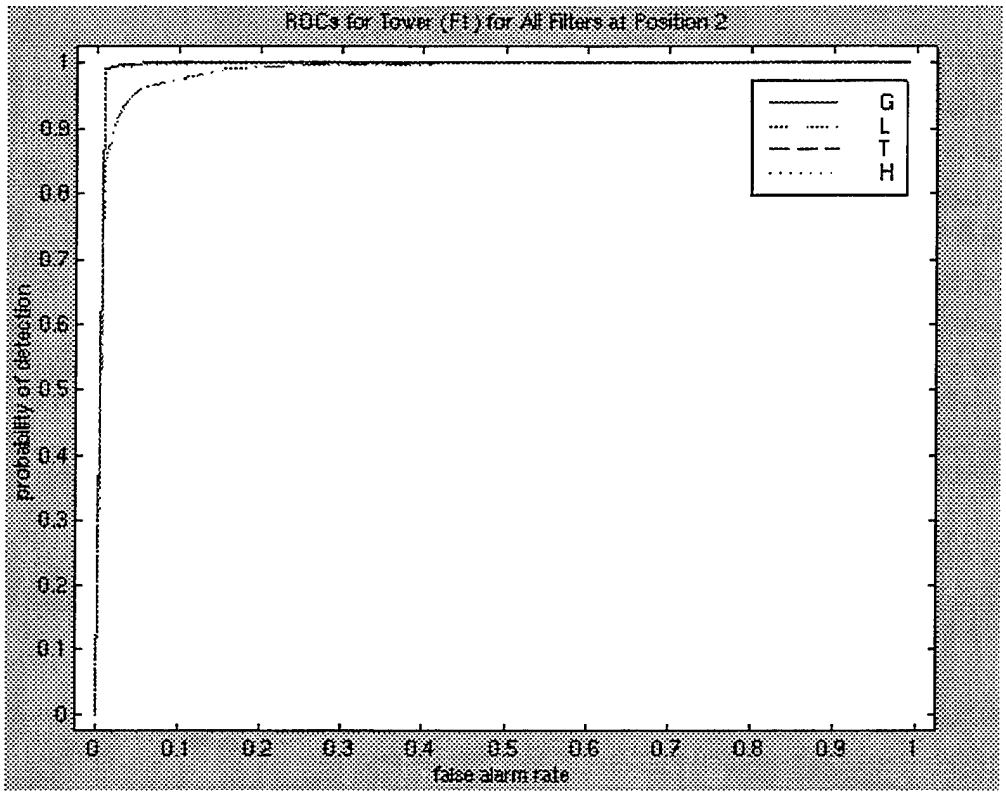

Sensitivities for Rectangle (IL) for All Filters at Position 2
probability of detection
false alarm rate
G
L
T
H

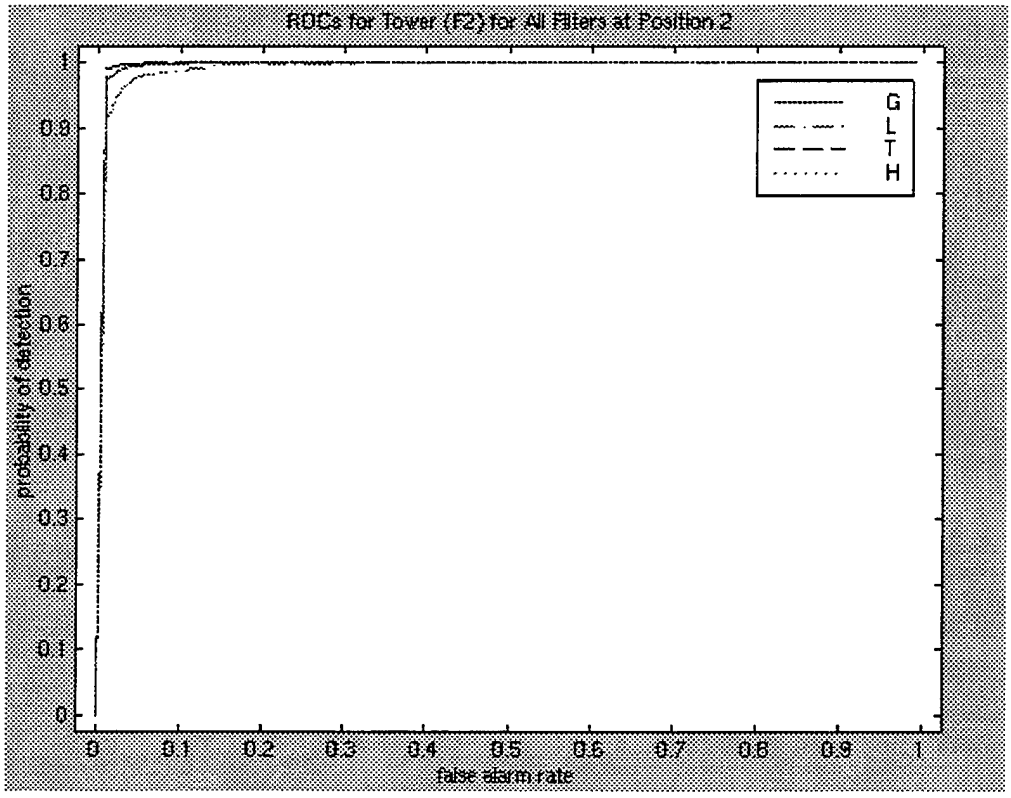

ROCs for Cylindrical Tank (F1) for All Filters at Position 2
probability of detection
false alarm rate
G
L
T
H

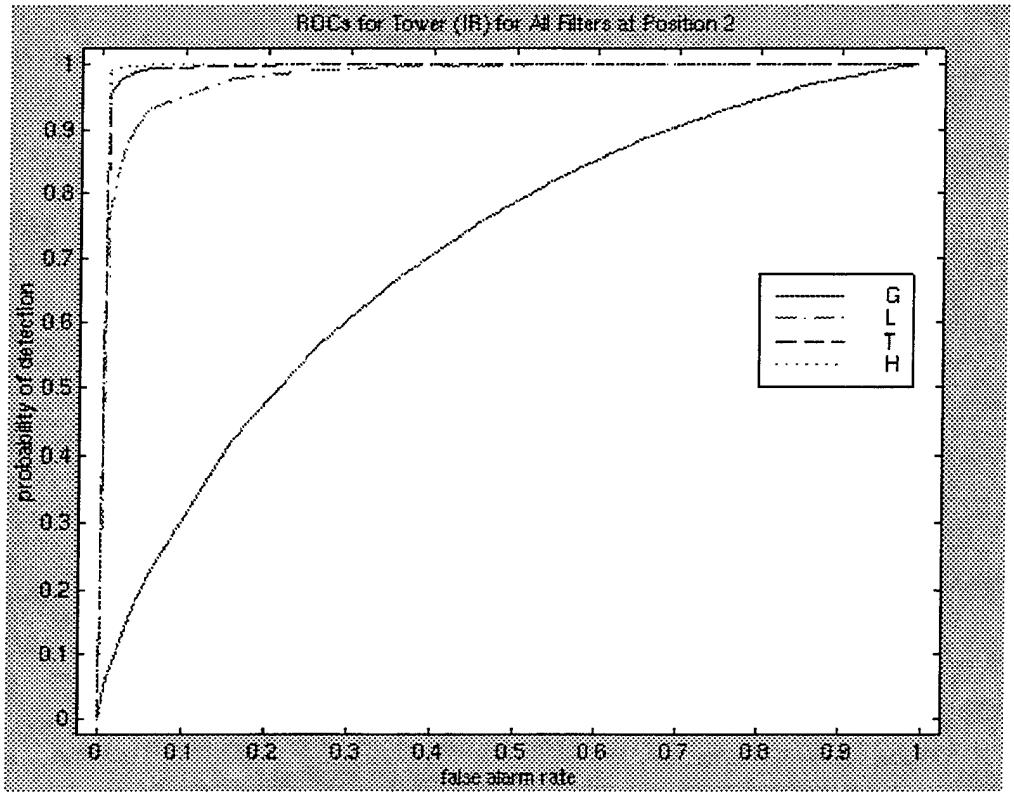

ROCs for Cylindrical Tank (F2) for All Filters at Position 2
probability of detection
false alarm rate
G
L
T
H

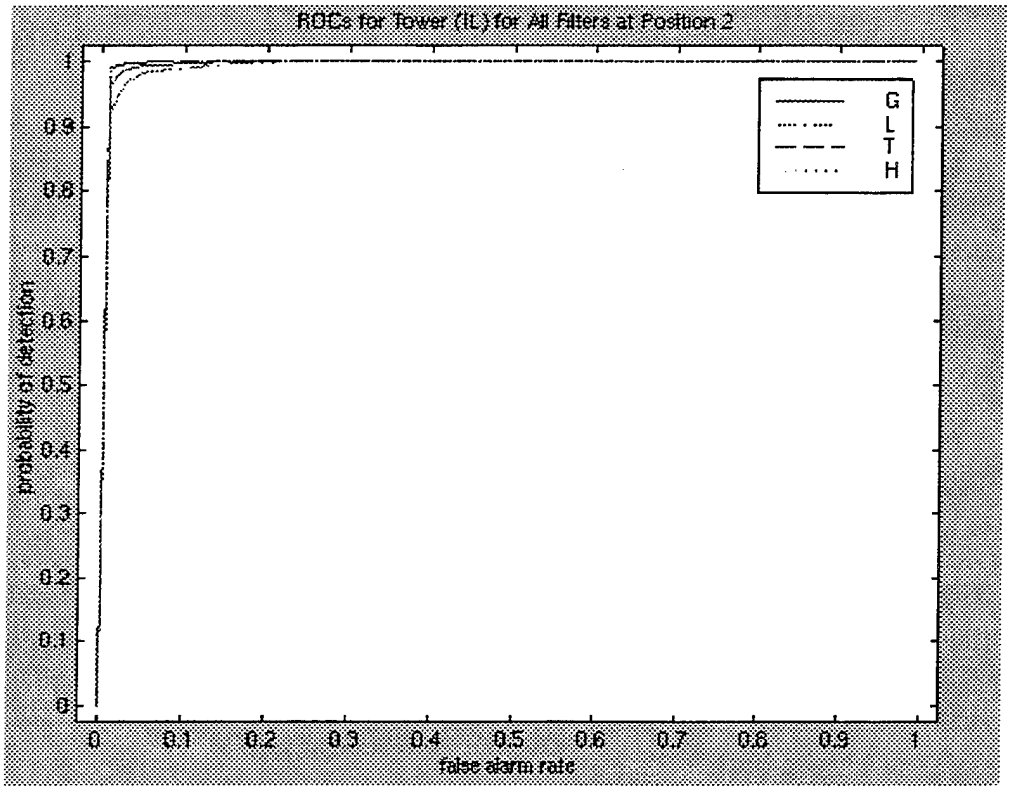

ROCs for Cylindrical Tank (IR) for All Filters at Position 2
probability of detection
false alarm rate
G
L
T
H

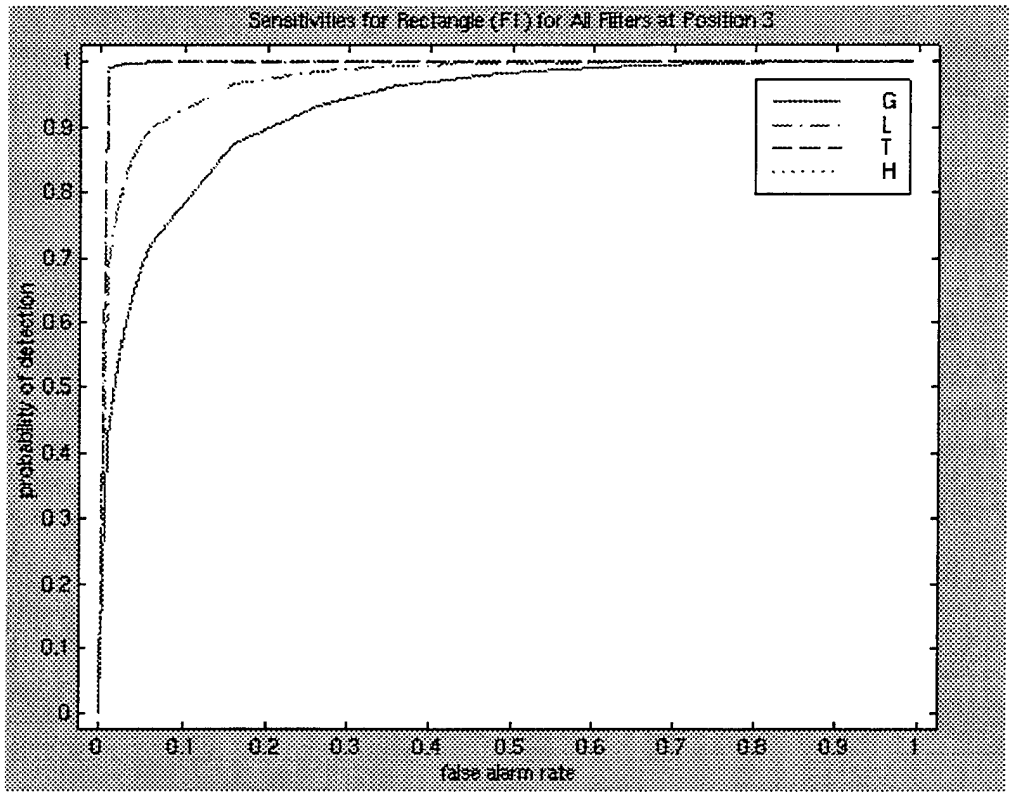

ROCs for Cylindrical Tank (IL) for All Filters at Position 2
probability of detection
false alarm rate
G
L
T
H

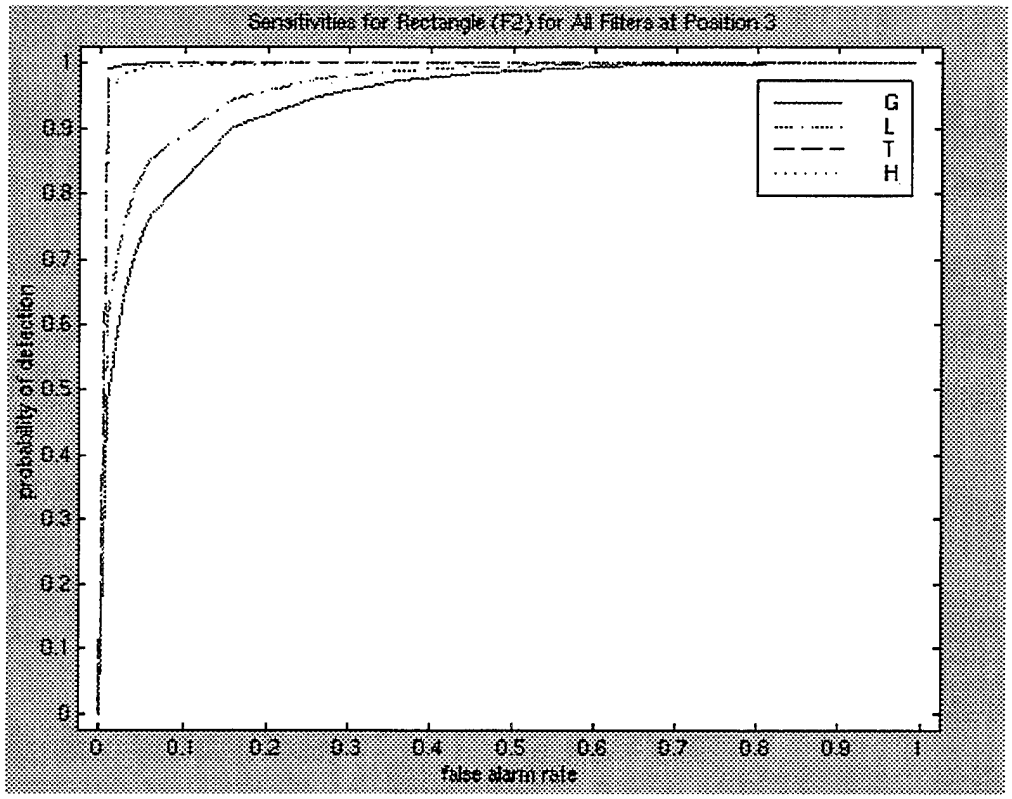

ROCs for Tower (F1) for All Filters at Position 2
probability of detection
false alarm rate
G
L
T
H

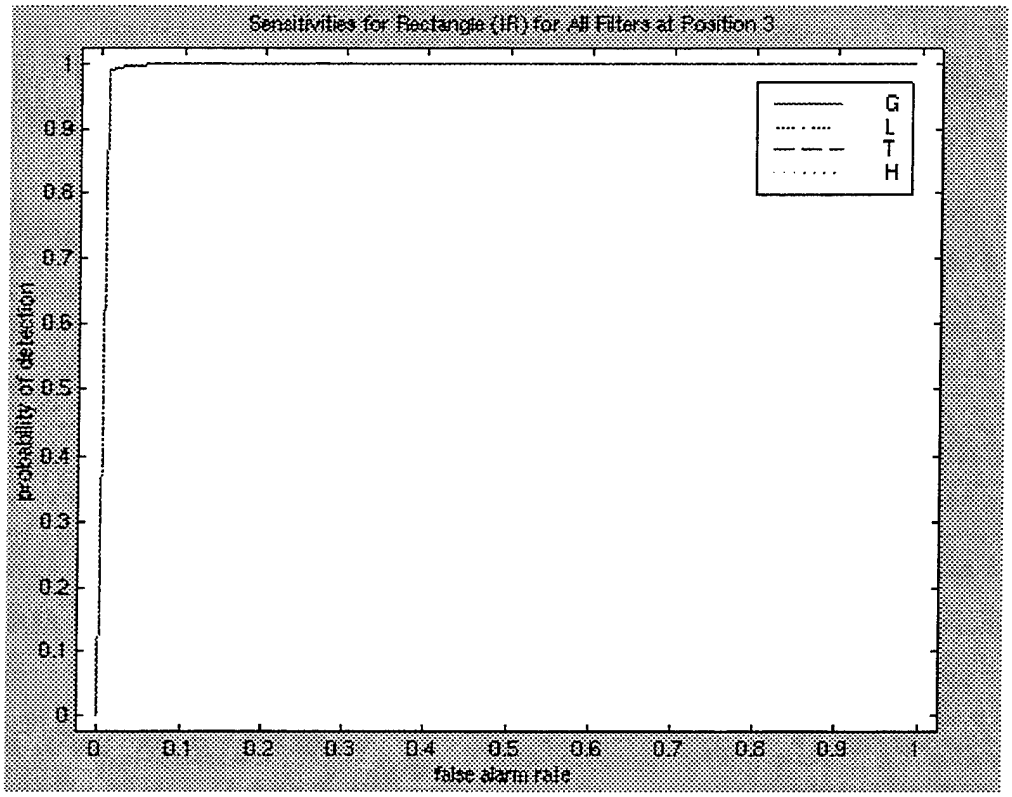

ROCs for Tower (F2) for All Filters at Position 2
probability of detection
false alarm rate
G
L
T
H

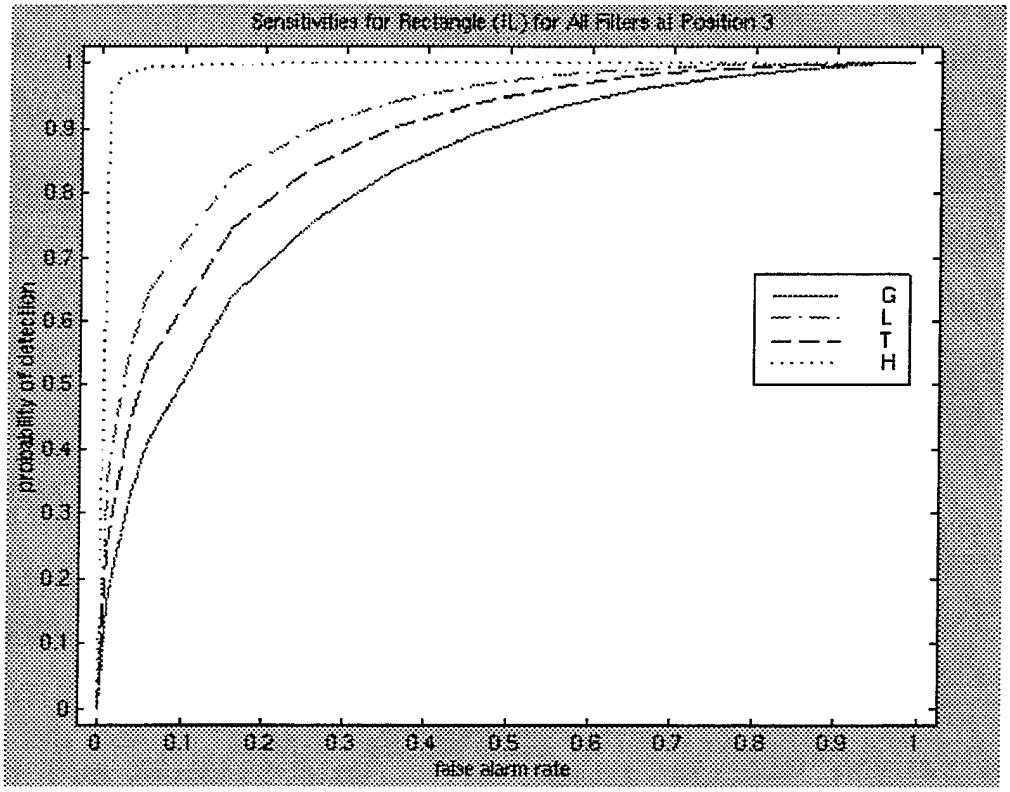

ROCs for Tower (IR) for All Filters at Position 2
probability of detection
false alarm rate
G
L
T
H

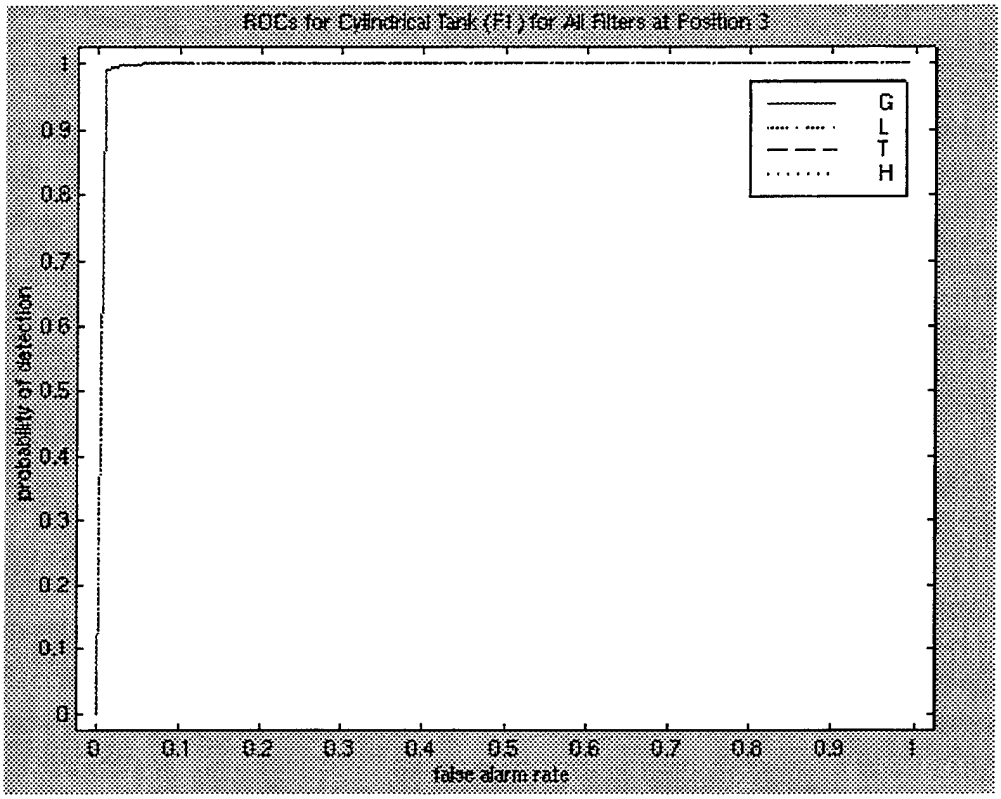

ROCs for Tower (IL) for All Filters at Position 2
probability of detection
false alarm rate
G
L
T
H

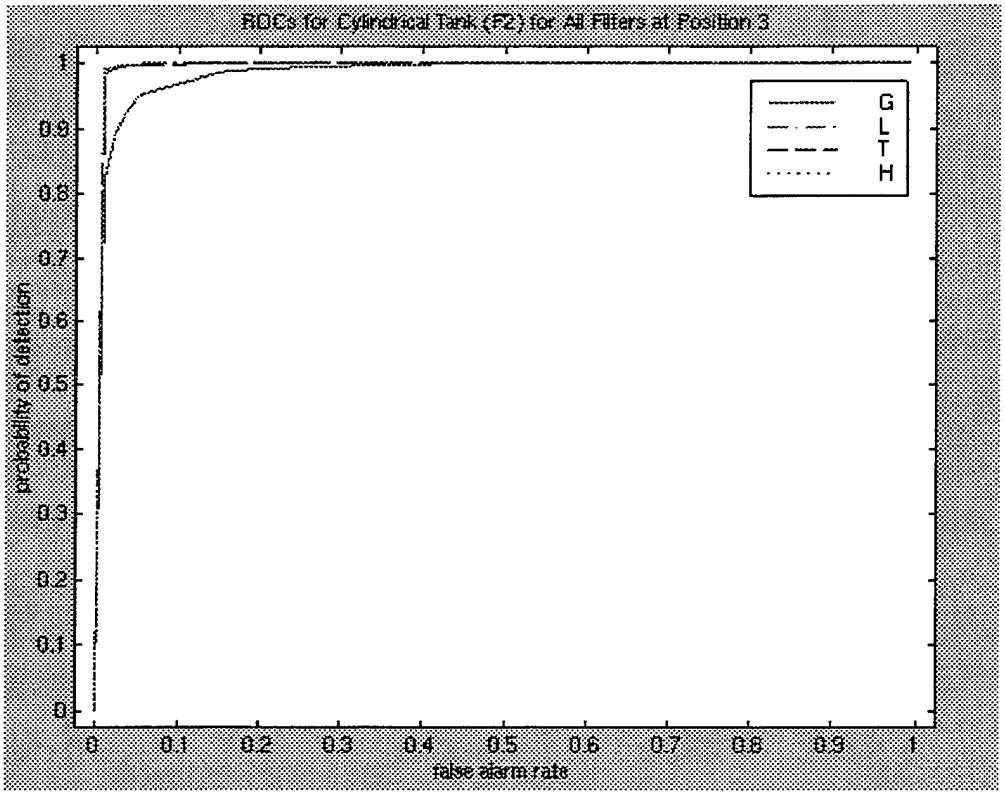

Sensitivities for Rectangle (F1) for All Filters at Position 3
probability of detection
false alarm rate
G
L
T
H

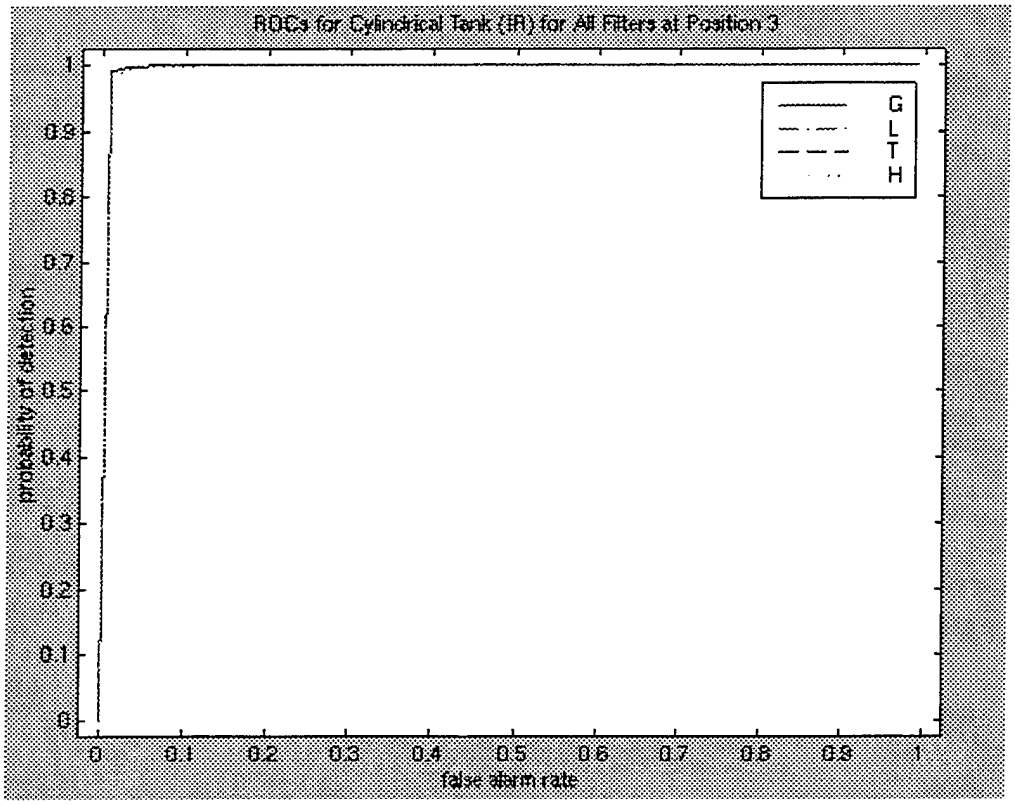

Sensitivities for Rectangle (F2) for All Filters at Position 3
probability of detection
false alarm rate
G
L
T
H

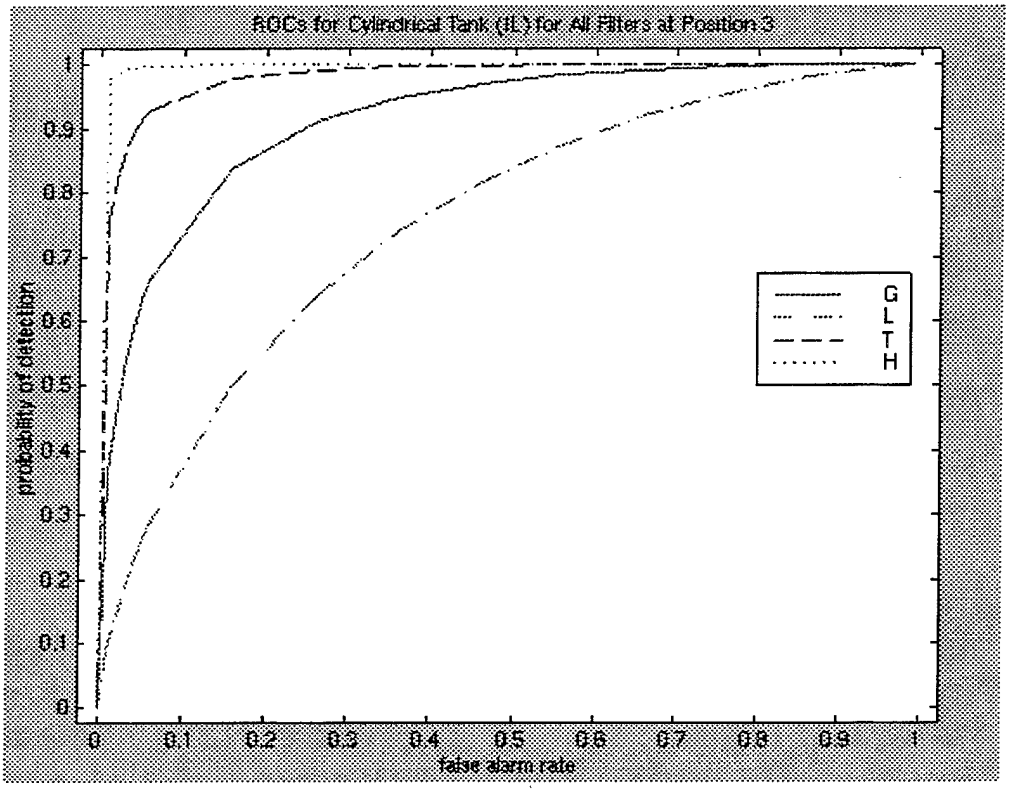

Sensitivities for Rectangle (IR) for All Filters at Position 3
probability of detection
false alarm rate
G
L
T
H

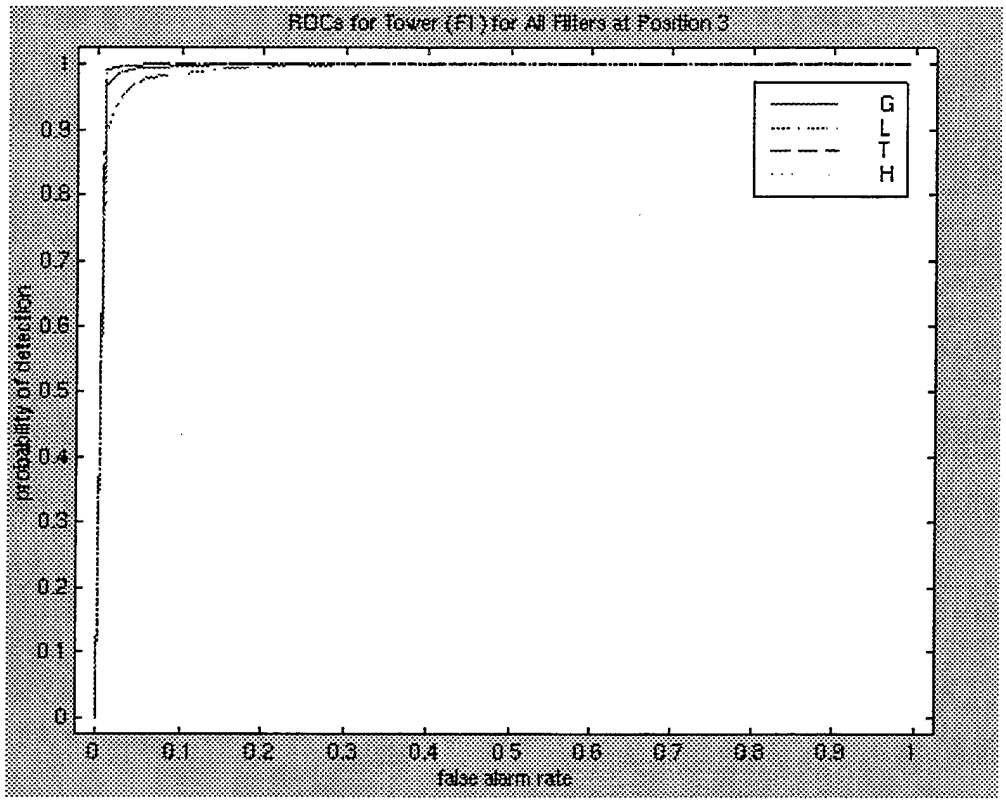

Sensitivities for Rectangle (IL) for All Filters at Position 3
probability of detection
false alarm rate
G
L
T
H

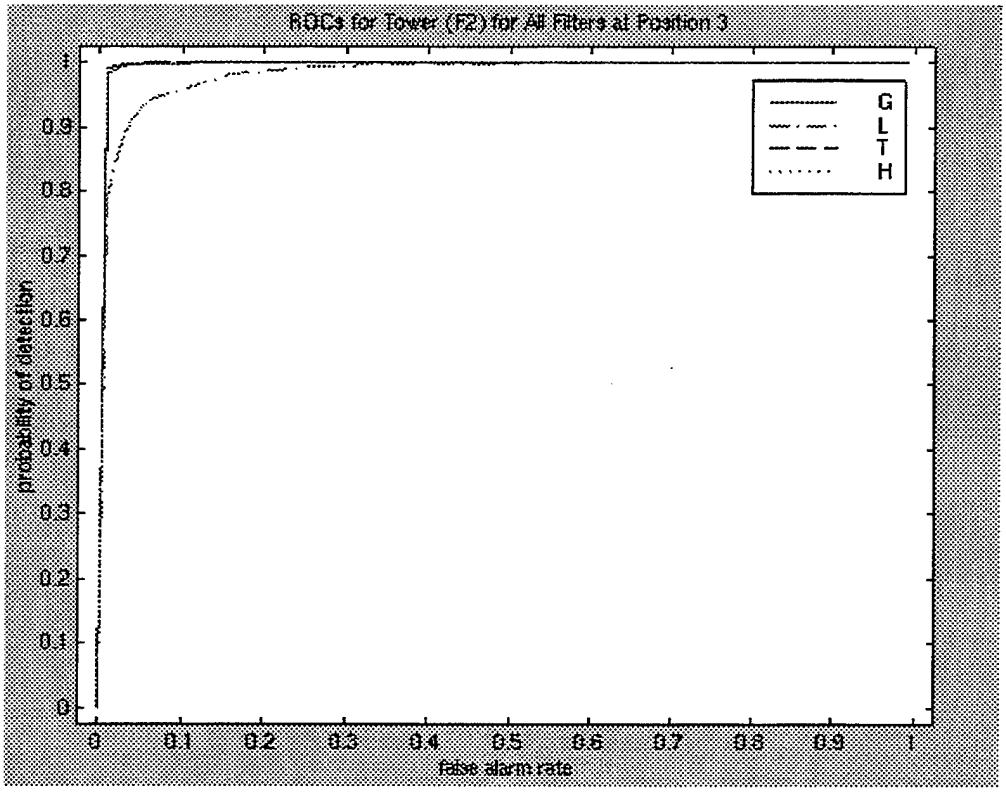

ROCs for Cylindrical Tank (F1) for All Filters at Position 3
probability of detection
false alarm rate
G
L
T
H

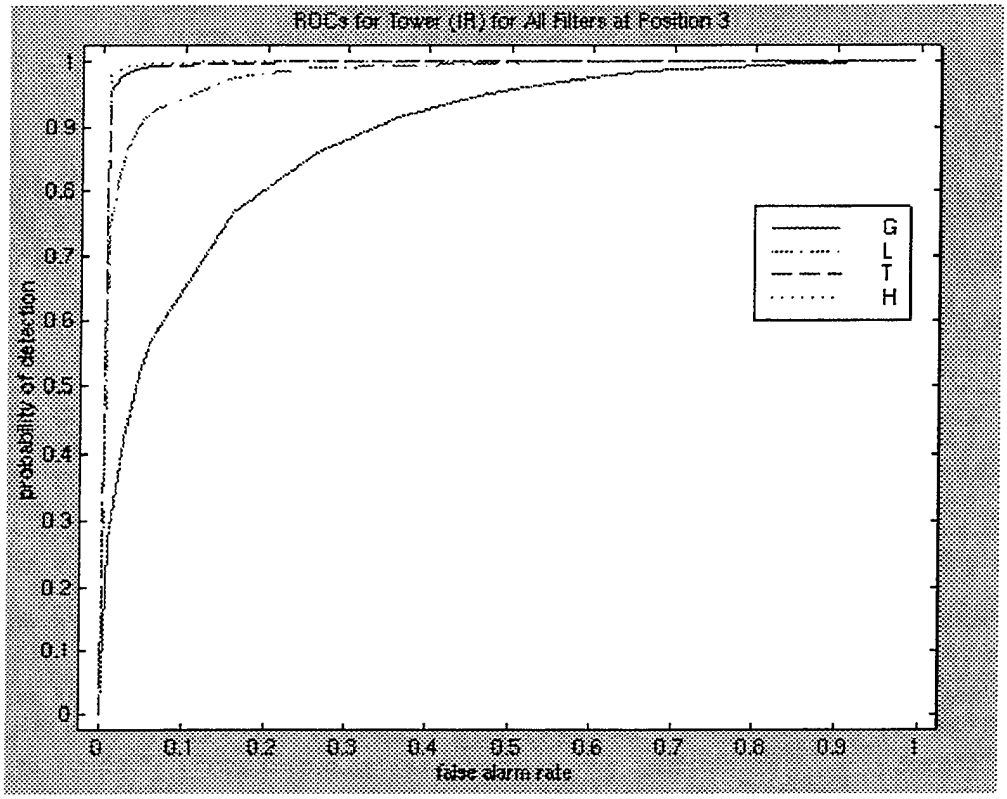

ROCs for Cylindrical Tank (F2) for All Filters at Position 3
probability of detection
false alarm rate
G
L
T
H

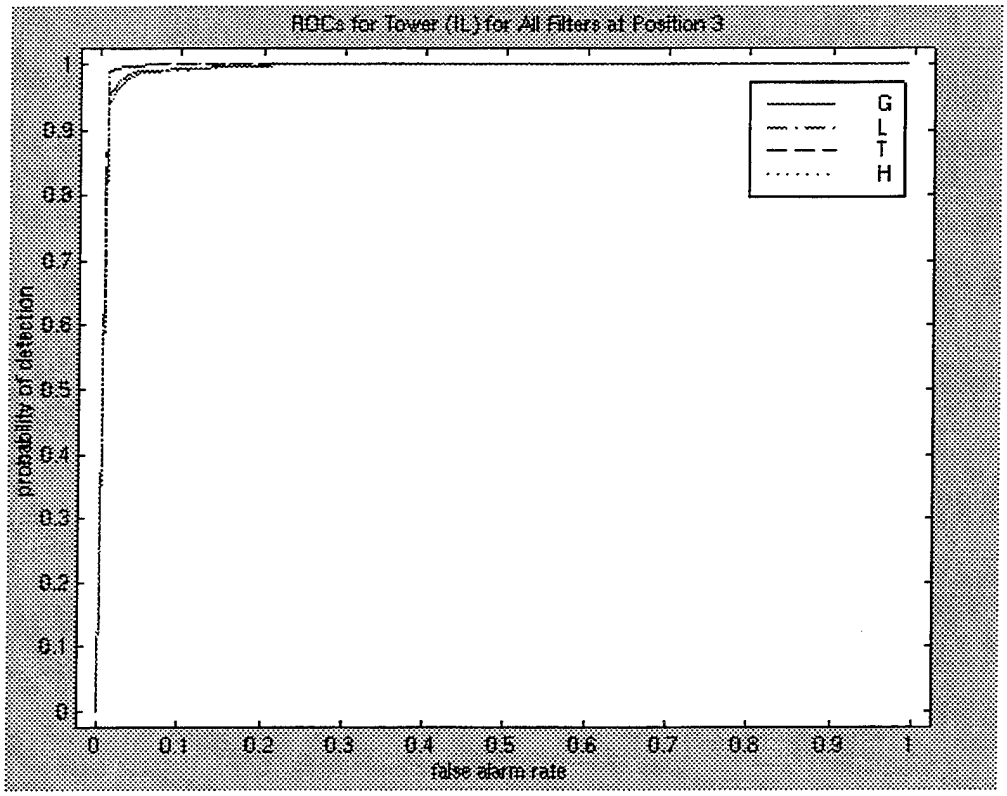

ROCs for Cylindrical Tank (IR) for All Filters at Position 3
probability of detection
false alarm rate
G
L
T
H


ROCs for Cylindrical Tank (IL) for All Filters at Position 3
probability of detection
false alarm rate
G
L
T
H


ROCs for Tower (F1) for All Filters at Position 3
probability of detection
false alarm rate
G
L
T
H


ROCs for Tower (F2) for All Filters at Position 3
probability of detection
false alarm rate
G
L
T
H


ROCs for Tower (IR) for All Filters at Position 3
probability of detection
false alarm rate
G
L
T
H


ROCs for Tower (IL) for All Filters at Position 3
probability of detection
false alarm rate
G
L
T
H

APPENDIX R. MATLAB FUNCTION FOR CENSORED REGRESSION

filename: **Censored RegrssnMLEs.m**

```
function [Coefs, StdDev, CovMatrx] = CensoredRegrssnMLEs (X, Y,
InitCoefs, InitStdDev, a, Tol,ShowCalc)

%  "X" contains columns of the independent variable- a column of ones is
used for
%  if an intercept coefficient is desired.
%  "Y" is a column of the response variable.
%  "InitCoefs" is a column of the estimated coefficient values in the
order of
%  the columns of "X" (A good start are the coefficient estimates from
OLS).
%  "InitStdDev" a scalar estimate of the standard deviation.
%  "a" is the censored value (for centored from above).
%  "Tol" is the error tolerance for terminating Newton's Method.
%  "ShowCalc" is a "Yes" or "No" indicator whether or not to show the
intermediate
%  calculations.

X=X';
Y=Y';
Censored=(Y==a);
NotCensored=(Y~=a);
Ngamma=length (InitCoefs);
NextGamma=InitCoefs/InitStdDev;
NextTheta=1/InitStdDev;
n=length(Y);
Error=1;
Gradnt1=[];
Gradnt2=[];
while Error>Tol
  Gamma=NextGamma;
  Theta=NextTheta;
  Gradnt_G=[0;0];
  Gradnt_T=(n^2)/(2*Theta);
  Gradnt_GG=[0,0;0,0];
  Gradnt_TG=[0;0];
  Gradnt_TT=(-n^2)/(2*Theta^2);
  for i=1:n
    Alpha(i)=a*Theta-(Gamma')*X(:,i);
    Epsilon(i)=Theta*Y(i)-(Gamma')*X(:,i);
    Lambda(i)=normpdf(Alpha(i))/normcdf(Alpha(i));
    Delta(i)=(Lambda(i)*Alpha(i)-Lambda(i)^2);
    Gradnt_G=Gradnt_G+NotCensored(i)*Epsilon(i)*X(:,i)-
Censored(i)*Lambda(i)*X(:,i);
    Gradnt_T=Gradnt_T-NotCensored(i)*Epsilon(i)*Y(i)+a*Lambda(i);
    Gradnt_GG=Gradnt_GG-NotCensored(i)*X(:,i)*(X(:,i)')-
Censored(i)*Delta(i)*X(:,i)*(X(:,i)');
    Gradnt_TG=Gradnt_TG+NotCensored(i)*Y(i)*X(:,i)-a*Delta(i)*X(:,i);
    Gradnt_TT=Gradnt_TT-NotCensored(i)*Y(i)^2+Censored(i)*Delta(i)*a^2;
  end;
  if ShowCalc=='Y'
    format short g
    disp('        Aphla      Epsilon       Lambda        Delta');
    disp([Alpha', Epsilon', Lambda', Delta']);
    format long g
  end;
  clear Alpha Epsilon Lambda Delta
  Grad1=[Gradnt_G; Gradnt_T];
```

```
    disp(' ')
    disp('Current Coefficients:')
    disp(Gamma/Theta)
    disp(' ')
    disp('Current Std Dev:')
    disp(1/Theta)
    disp(' ')
    Error=sum(Grad1.^2)
    disp(' ')
    disp(' ')
    disp(' ')
    disp(' ')
    disp(' ')
    Grad2=[Gradnt_GG, Gradnt_TG];
    Grad2=[Grad2; Gradnt_TG', Gradnt_TT];
    NextParams=[Gamma; Theta] + (inv(-Grad2)*Grad1);
    NextGamma=NextParams(1:(Ngamma),1);
    NextTheta=NextParams((Ngamma+1),1);
end;
disp(' ')
disp('GradF1:')
disp(Grad1)
disp(' ')
disp('Hessian:')
disp(Grad2)
Coefs=Gamma/Theta;
StdDev=1/Theta;
J=eye(Ngamma);
J=[J/Theta, (-Gamma/(Theta^2))];
J=[J; zeros(1,Ngamma), (-1/(Theta^2))]
V=inv(-Grad2)
CovMatrx=J*V*J'
```

APPENDIX S: OUTPUT OF CENSORED REGRESSION FUNCTION

```
<<<<<<<<<<<<<<<<<<<<<<<<<<<<  H~Time  >>>>>>>>>>>>>>>>>>>>>>>>
        Aphla       Epsilon        Lambda         Delta
       11.984     -0.019406    2.6109e-32    3.1288e-31
       10.071       -1.9322    3.7766e-23    3.8034e-22
        11.85       -0.1534    1.2889e-31    1.5273e-30
       11.764       -9.8874    3.5347e-31    4.1583e-30
       4.7565        4.7565    4.8766e-06    2.3195e-05
       6.7717       0.77014    4.4004e-11    2.9798e-10
       3.1006        -2.901     0.0032643       0.01011
       7.0372       -4.8163    7.0343e-12    4.9502e-11
       19.855        13.853    1.0015e-86    1.9885e-85
       9.5185        3.5169    8.4558e-21    8.0486e-20
       33.105       -8.3627    4.114e-239    1.362e-237
       7.1688        1.1673     2.763e-12    1.9807e-11
       9.8947        3.8932    2.1934e-22    2.1703e-21
       20.325        -2.727    7.8763e-91    1.6009e-89
       11.063        5.0617    1.0549e-27    1.1671e-26
       12.266       -8.7022    8.4913e-34    1.0416e-32
       3.3326        3.3326     0.0015466     0.0051519
       3.1996        3.1996     0.0023888     0.0076375
       1.5446        1.5446        0.1289       0.18249
       6.3736        6.3736     6.023e-10    3.8388e-09
       3.8652       -2.1364    0.00022743      0.000879
       4.3002       -1.7014     3.851e-05     0.0001656
       8.1179        8.1179     1.954e-15    1.5862e-14
        2.219         2.219      0.034476      0.075313
       12.727        12.727     2.683e-36    3.4146e-35
       15.821        3.8177    1.7759e-55    2.8096e-54
       8.6888        8.6888     1.612e-17    1.4006e-16
       11.281     -0.57233    9.2386e-29    1.0422e-27
       5.2245        5.2245    4.7186e-07    2.4652e-06
        3.219         3.219     0.0022444     0.0072196
       4.6822       -1.3193    6.9239e-06    3.2419e-05
       11.535        5.5331    5.1297e-30    5.9169e-29
       7.4839        7.4839    2.7473e-13     2.056e-12
       4.8686        4.8686    2.8427e-06     1.384e-05
        13.79        7.7881    2.0401e-42    2.8133e-41
       3.6365        3.6365    0.00053623     0.0019497

Current Coefficients:
          5.98263368946008
      -0.00255997229609403

Current Std Dev:
        0.0527320088145204

Error = 5.95730258181792e-11

GradF1:
     -1.38214264967969e-08
     -7.71834255025583e-06
     -4.80242221293989e-09

Hessian:
   -22.2797993283313       -16756.6236532761          87.776141079539
-16756.6236532761       -13010602.3554909          65479.3849930979
    87.776141079539          65479.3849930979        -361.259019184772
```

```
J =
0.0527320088145204                          0      -0.315476292446656
                 0        0.0527320088145204       0.000134992481682558
                 0                          0      -0.00278066475361465
V =
 3.08028571493132       -0.00228384901163305       0.334470721458924
-0.00228384901163305     2.56880468325853e-06     -8.9309056684917e-05
 0.334470721458924      -8.93090566849174e-05      0.0678478482918518

CovMatrx =
 0.00418951052182767     -5.3734266455122e-06      1.04749715572985e-05
-5.3734266455122e-06      7.10789435701439e-09     -1.2372577423993e-08
 1.04749715572985e-05    -1.23725774239929e-08      5.24606108409868e-07
```

<<<<<<<<<<<<<<<<<<<<<<<<<<<<< G~Time >>>>>>>>>>>>>>>>>>>>>>>>>>>>

Aphla	Epsilon	Lambda	Delta
9.3536	2.0218	4.0046e-20	3.7458e-19
8.4849	-4.194	9.2848e-17	7.878e-16
9.2928	9.2928	7.0628e-20	6.5633e-19
9.254	-4.8185	1.0122e-19	9.3669e-19
6.0711	6.0711	3.9558e-09	2.4016e-08
6.9864	1.4748	1.0047e-11	7.0193e-11
5.319	5.319	2.8672e-07	1.5251e-06
7.107	4.343	4.2946e-12	3.0522e-11
12.929	7.2339	2.0196e-37	2.6111e-36
8.234	5.9106	7.5639e-16	6.2281e-15
18.947	5.8333	4.4393e-79	8.4111e-78
7.1668	4.6448	2.8036e-12	2.0093e-11
8.4048	-8.2486	1.8253e-16	1.5341e-15
13.142	-1.0504	1.2465e-38	1.6382e-37
8.9356	8.9356	1.8318e-18	1.6368e-17
9.482	-7.6616	1.1959e-20	1.1339e-19
5.4244	5.4244	1.6277e-07	8.8293e-07
5.364	4.873	2.2548e-07	1.2095e-06
4.6123	4.6123	9.5824e-06	4.4197e-05
6.8056	1.2073	3.4957e-11	2.379e-10
5.6663	5.6663	4.2564e-08	2.4118e-07
5.8638	3.8697	1.3627e-08	7.9905e-08
7.5978	-11.057	1.1632e-13	8.838e-13
4.9186	2.3966	2.2265e-06	1.0951e-05
9.6912	-2.4947	1.6087e-21	1.5591e-20
11.096	-0.39358	7.2998e-28	8.1002e-27
7.8571	7.8571	1.5683e-14	1.2323e-13
9.0346	-6.9966	7.5258e-19	6.7992e-18
6.2837	6.2837	1.0641e-09	6.6863e-09
5.3728	-1.3872	2.1505e-07	1.1554e-06
6.0374	6.0374	4.8521e-09	2.9294e-08
9.1497	-3.7751	2.6428e-19	2.4181e-18
7.3099	4.8853	9.951e-13	7.274e-12
6.122	5.126	2.8998e-09	1.7753e-08
10.174	-4.0337	1.3318e-23	1.3549e-22
5.5624	2.2914	7.6255e-08	4.2416e-07

Current Coefficients:
```
          6.21068116457626
      -0.00455346930467687
```

Current Std Dev:
```
         0.206509463879223
```

Error = 3.79556873919512e-23

GradF1:
```
                         0
     -6.13908923696727e-12
     -5.16982332874014e-13
```

Hessian:
```
   -26.0000469060378        -19034.6359576193        74.8039173755412
-19034.6359576193        -14405957.8923267        52620.7158448038
    74.8039173755412         52620.7158448038       -280.245923135296
```

```
J =
0.206509463879223                          0      -1.28256443762143
                0          0.206509463879223       0.000940334504899319
                0                          0      -0.042646158671684
V =
 1.87497616538216       -0.00206697555620314        0.112364984307132
-0.00206697555620314     2.49960274112808e-06      -8.23811561624158e-05
 0.112364984307132      -8.23811561624158e-05       0.018092629273247

CovMatrx =
 0.0502000309617088      -6.63294535171942e-05     2.26118834356802e-08
-6.63294535171943e-05     9.06016225621104e-08    -2.70887177893557e-11
 2.26118834355422e-08    -2.7088717789066e-11      3.29049716722707e-05
```

<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<< L~Time >>>>>>>>>>>>>>>>>>>>>>>>>>>>

Aphla	Epsilon	Lambda	Delta
10.302	-12.109	3.5986e-24	3.7072e-23
10.063	-1.6977	4.0958e-23	4.1215e-22
10.285	4.6817	4.2749e-24	4.3967e-23
10.274	-1.8374	4.7704e-24	4.9013e-23
9.3992	9.3992	2.6115e-20	2.4546e-19
9.6509	9.6509	2.3765e-21	2.2935e-20
9.1925	9.1925	1.7852e-19	1.641e-18
9.684	-7.8264	1.7247e-21	1.6702e-20
11.285	-1.8995	8.8946e-29	1.0037e-27
9.9939	1.8233	8.1808e-23	8.1758e-22
12.939	5.0156	1.7598e-37	2.277e-36
9.7005	2.2926	1.4708e-21	1.4268e-20
10.041	-0.88917	5.1098e-23	5.1307e-22
11.343	1.0901	4.5761e-29	5.1908e-28
10.187	9.844	1.1681e-23	1.1899e-22
10.337	-4.3601	2.5004e-24	2.5847e-23
9.2214	9.2214	1.3671e-19	1.2607e-18
9.2048	9.2048	1.5932e-19	1.4665e-18
8.9982	8.9982	1.045e-18	9.4035e-18
9.6012	-8.2671	3.8349e-21	3.6819e-20
9.2879	2.6226	7.388e-20	6.8619e-19
9.3423	4.3044	4.4544e-20	4.1614e-19
9.819	1.5231	4.6263e-22	4.5425e-21
9.0824	4.3573	4.8814e-19	4.4335e-18
10.395	0.92266	1.3773e-24	1.4316e-23
10.781	0.16462	2.3045e-26	2.4845e-25
9.8903	9.8903	2.2915e-22	2.2664e-21
10.214	-3.8281	8.8494e-24	9.0387e-23
9.4577	9.4577	1.5052e-20	1.4236e-19
9.2073	9.2073	1.558e-19	1.4345e-18
9.39	9.39	2.8492e-20	2.6754e-19
10.246	-8.7017	6.4021e-24	6.5594e-23
9.7398	4.1997	1.0034e-21	9.7731e-21
9.4132	1.4834	2.2892e-20	2.1549e-19
10.527	1.9015	3.4369e-25	3.6181e-24
9.2594	5.1899	9.6277e-20	8.9147e-19

Current Coefficients:
3.56042631224946
-0.0011707103015085

Current Std Dev:
0.193119413640914

Error = 1.07203388997578e-21

GradF1:
-3.37507799486048e-14
-3.27418092638254e-11
8.1712862560022e-14

Hessian:

-26	-19314.5	69.9594
-19314.5	-14786409.3522	51457.252257
69.9594	51457.252257	-237.178213115407

```
J =
0.193119413640914                          0      -0.687587441733296
                0          0.193119413640914       0.000226086886970699
                0                          0      -0.0372951079250102
V =
 1.55969438215457       -0.00178097726166561        0.0736622757710664
-0.00178097726166561     2.30971136831878e-06      -2.42210279092277e-05
 0.0736622757710661     -2.42210279092273e-05       0.0206891729559566

CovMatrx =
 0.0483876157407601      -6.32055140611848e-05       9.05753789592455e-18
-6.32055140611848e-05     8.5083401878941e-08       -1.2762434814291e-20
 1.12757025938492e-17    -1.50083650341621e-20       2.87770894483103e-05
```

<<<<<<<<<<<<<<<<<<<<<<<<<<<<< T~Time >>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Aphla	Epsilon	Lambda	Delta
9.5468	9.5468	6.4566e-21	6.164e-20
9.4357	9.4357	1.8519e-20	1.7474e-19
9.539	9.539	6.954e-21	6.6334e-20
9.534	-10.657	7.2908e-21	6.9511e-20
9.1273	9.1273	3.2433e-19	2.9602e-18
9.2442	8.1697	1.1076e-19	1.0239e-18
9.0312	9.0312	7.7617e-19	7.0098e-18
9.2597	-5.1526	9.6038e-20	8.8928e-19
10.004	10.004	7.4231e-23	7.4258e-22
9.4037	9.4037	2.505e-20	2.3556e-19
10.773	6.4979	2.5165e-26	2.7109e-25
9.2673	9.2673	8.948e-20	8.2923e-19
9.4255	9.4255	2.0395e-20	1.9223e-19
10.031	10.031	5.6468e-23	5.6643e-22
9.4933	9.4933	1.0737e-20	1.0193e-19
9.5632	-9.6499	5.5202e-21	5.279e-20
9.0446	9.0446	6.8722e-19	6.2156e-18
9.0369	7.9624	7.369e-19	6.6593e-18
8.9409	8.9409	1.7474e-18	1.5623e-17
9.2211	-3.9064	1.3709e-19	1.2642e-18
9.0755	9.0755	5.1936e-19	4.7135e-18
9.1008	9.1008	4.1288e-19	3.7575e-18
9.3224	5.0476	5.3624e-20	4.999e-19
8.98	8.98	1.2305e-18	1.105e-17
9.5899	9.5899	4.2731e-21	4.0978e-20
9.7695	9.7695	7.5135e-22	7.3403e-21
9.3555	9.3555	3.9351e-20	3.6815e-19
9.506	-10.378	9.5212e-21	9.0508e-20
9.1544	9.1544	2.5302e-19	2.3162e-18
9.038	7.9635	7.2943e-19	6.5926e-18
9.123	9.123	3.3734e-19	3.0775e-18
9.5207	-1.2741	8.2779e-21	7.8811e-20
9.2856	9.2856	7.5518e-20	7.0123e-19
9.1338	9.1338	3.0561e-19	2.7914e-18
9.6516	5.3768	2.3606e-21	2.2784e-20
9.0623	9.0623	5.8579e-19	5.3086e-18

Current Coefficients:
```
          3.53892707621323
     -0.000424821448308802
```

Current Std Dev:
```
         0.150767399551903
```

Error = 6.78351626339064e-17

GradF1:
```
      1.06575552500295e-14
      8.18510411282111e-12
      8.23620638630817e-09
```

Hessian:
```
  -12                -8880.26              38.6946
-8880.26          -6861224.772           28511.797113
   38.6946           28511.797113          -154.284072861433
```

```
J =
0.150767399551903                         0     -0.533554832484488
                0          0.150767399551903      6.40492250353914e-05
                0                         0     -0.0227308087676432
V =
 2.39919865995144       -0.0026060046643987          0.120130068235466
-0.0026060046643869      3.45868842729863e-06       -1.4420706749299e-05
 0.120130068235466      -1.44207067492995e-05         0.0339453358739479

CovMatrx =
 0.0448721419447545    -5.80765534533235e-05      8.67496687137743e-19
-5.80765534533235e-05   7.8479531166867e-08      -1.03970217297226e-21
-3.79470760369927e-19   6.28657265540301e-22      1.75392042967603e-05
```

<<<<<<<<<<<<<<<<<<<<<<<<<<<<< H~CSFIQM >>>>>>>>>>>>>>>>>>>>>>>>>>>>

Aphla	Epsilon	Lambda	Delta
3.4087	-2.4791	0.0011966	0.0040774
3.4206	-2.4672	0.0011489	0.0039287
3.3973	-2.4906	0.0012441	0.0042249
3.4408	-7.1799	0.001072	0.0036873
3.4106	3.4106	0.0011888	0.0040531
3.4305	0.48653	0.0011107	0.0038092
3.4203	0.47631	0.0011503	0.0039331
3.4141	-2.4004	0.0011747	0.0040092
3.4221	0.47811	0.0011433	0.003911
3.4359	0.49194	0.0010903	0.003745
3.4192	-16.922	0.0011546	0.0039465
3.4429	0.49892	0.0010644	0.0036636
3.4156	0.47162	0.0011689	0.0039912
3.3849	-7.9227	0.0012974	0.00439
3.4276	0.48365	0.0011218	0.0038438
3.4394	-6.8462	0.0010773	0.0037041
3.4281	3.428	0.0011199	0.003838
3.4141	3.4141	0.0011747	0.0040092
3.4103	3.4102	0.0011902	0.0040576
3.4645	3.4645	0.00098776	0.0034211
3.3653	0.42136	0.0013862	0.0046631
3.4236	0.47968	0.0011372	0.0038919
3.4033	3.4033	0.0012189	0.0041467
3.3908	3.3908	0.0012718	0.0043107
3.428	3.4279	0.0011204	0.0038395
3.3984	-2.4895	0.0012395	0.0042107
3.4116	3.4116	0.0011849	0.0040409
3.4531	-2.3614	0.0010276	0.0035473
3.4393	3.4392	0.0010777	0.0037055
3.429	3.429	0.0011163	0.0038264
3.4063	0.46236	0.0012065	0.0041081
3.4295	0.48557	0.0011144	0.0038207
3.4146	3.4146	0.0011728	0.0040032
3.4399	3.4398	0.0010755	0.0036985
3.4141	0.47018	0.0011747	0.0040092
3.3694	3.3694	0.0013672	0.0046049

Current Coefficients:
9.09762576101784
-12.9266899736184

Current Std Dev:
0.107501011896347

Error = 1.01525409396449e-11

GradF1:
7.66089556236693e-07
2.85194845517367e-07
3.07966096048905e-06

Hessian:

-36	-13.402253	153.603951914641
-13.402253	-4.989456220079	57.1844110631026
153.603951914641	57.1844110631026	-676.068838723457

```
J =
0.107501011896347                          0        -0.978003975163692
                0          0.107501011896347         1.38963225263435
                0                          0        -0.0115564675587385
V =
 129777.49518052          -348593.68720298          0.312728942961257
-348593.68720298           936359.644683127        -0.285684537215275
 0.312728942970085        -0.28568453724249         0.0483674111766335

CovMatrx =
 1499.74991753792      -4028.50061839228            0.000158148439339712
-4028.50061839228      10821.0179032868            -0.000421828836432389
 0.000158148439328746    -0.000421828836398579      6.45956171324861e-06
```

```
<<<<<<<<<<<<<<<<<<<<<<<<<<<<  H~G  >>>>>>>>>>>>>>>>>>>>>>>>>>>>
        Aphla       Epsilon        Lambda         Delta
       4.5505       -2.5211   1.2719e-05   5.7879e-05
         6.67      -0.40166   8.7168e-11   5.8141e-10
       1.6442       -5.4274      0.10869      0.16689
       7.2224       -5.5337   1.8787e-12   1.3569e-11
       1.6442        1.6442      0.10869      0.16689
        3.829       0.29311   0.00026144     0.001001
       1.6442       -1.8916      0.10869      0.16689
       2.7398       -4.2437    0.0093795      0.02561
       3.9015       0.36567    0.0001975   0.00077051
       2.5652      -0.97067     0.014939     0.038097
       6.8423       -17.589   2.7202e-11   1.8612e-10
       2.6439      -0.89197     0.012157     0.031994
       8.2455        4.7096   6.8791e-16   5.6721e-15
         7.27       -6.3111   1.3306e-12   9.6736e-12
       1.6442       -1.8916      0.10869      0.16689
       8.4397       -3.9138   1.3606e-16   1.1483e-15
       1.6442        1.6442      0.10869      0.16689
       1.8388        1.8388     0.076072       0.1341
       1.6442        1.6442      0.10869      0.16689
       3.8633        3.8633   0.00022907   0.00088493
       1.6442       -1.8916      0.10869      0.16689
       2.4346       -1.1012      0.02075     0.050089
        9.039        9.0389   7.2321e-19   6.5371e-18
       2.6439        2.6438     0.012157     0.031994
       6.4746        6.4745   3.1484e-10   2.0385e-09
       6.1987      -0.87289   1.8079e-09   1.1207e-08
       1.6442        1.6442      0.10869      0.16689
       7.9988        1.0153      5.1e-15   4.0794e-14
       1.6442        1.6442      0.10869      0.16689
       4.3238        4.3237   3.4778e-05   0.00015037
       1.6442       -1.8916      0.10869      0.16689
       6.7675        3.2316   4.5272e-11   3.0638e-10
       2.6053        2.6052     0.013459     0.034884
        2.039         2.039     0.050954       0.1013
       7.2759        3.7401   1.2743e-12   9.2715e-12
       2.9408        2.9408    0.0052924     0.015536

 Current Coefficients:
         3.70618334057552
        0.171803772335592

 Current Std Dev:
       0.0895055145543749

 Error = 7.87845369785913e-11

GradF1:
     -1.37111956632907e-06
     -6.28141739333898e-06
     -6.11950681379467e-06

 Hessian:
 -36                  -121.3312              144.332468083947
-121.3312             -469.08840062          484.750954950162
 144.332468083947      484.750954950162     -673.771529409291
```

```
J =
0.0895055145543749                         0       -0.331723846931064
                 0       0.0895055145543749       -0.0153773850452798
                 0                         0       -0.00801123713564342
V =
 0.393768787744777      -0.0572341316549664       0.0431738649444601
-0.0572341316549663       0.0166294378087581      -0.000296254165708101
 0.0431738649444602     -0.000296254165708113      0.0105195614315077

CovMatrx =
 0.00174839489254475    -0.000455482138818791      -3.0018875512673e-06
-0.000455482138818791    0.000136525372203802       1.50835355028727e-06
-3.00188755126733e-06    1.50835355028727e-06       6.75144615774804e-07
```

```
<<<<<<<<<<<<<<<<<<<<<<<<<  H~L  >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
        Aphla       Epsilon        Lambda         Delta
       8.2702        1.6451   5.6064e-16   4.6367e-15
       5.1446       -1.4805   7.1382e-07   3.6724e-06
       3.3376       -3.2875    0.0015209     0.005074
       5.2477       -6.7031   4.1791e-07   2.1931e-06
       1.6932        1.6932      0.09964      0.15878
       1.6932       -1.6194      0.09964      0.15878
       1.6932       -1.6194      0.09964      0.15878
       6.8321       0.28942   2.9183e-11   1.9938e-10
       5.5624        2.2498   7.6264e-08   4.2421e-07
       4.0911       0.77846   9.2586e-05   0.00037877
       4.0186        -18.87   0.00012422   0.00049918
       3.8672       0.55462   0.00022563    0.0008725
       4.9009        1.5883   2.4283e-06   1.1901e-05
       4.7023       -8.0214    6.302e-06   2.9634e-05
       1.7938       -1.5188     0.082847      0.14175
       6.0064       -5.5672   5.8456e-09   3.5111e-08
       1.6932        1.6932      0.09964      0.15878
       1.6932        1.6932      0.09964      0.15878
       1.6932        1.6932      0.09964      0.15878
       6.9371         6.937   1.4163e-11   9.8251e-11
       3.6493        0.3367   0.00051181    0.0018675
       3.1717      -0.14092    0.0026111    0.0082747
       4.1279        4.1278     7.96e-05   0.00032857
       3.0799        3.0799    0.0034797     0.010705
        4.473        4.4729   1.8045e-05   8.0715e-05
       4.8088       -1.8164   3.7972e-06    1.826e-05
       1.6932        1.6932      0.09964      0.15878
       5.8142      -0.72844   1.8208e-08   1.0587e-07
       1.6932        1.6932      0.09964      0.15878
       1.6932        1.6932      0.09964      0.15878
       1.6932       -1.6194      0.09964      0.15878
       7.2538        3.9411   1.4972e-12    1.086e-11
       3.3191        3.3191    0.0016178     0.005367
       4.0204        4.0204   0.00012331   0.00049576
       4.2247       0.91204    5.313e-05   0.00022445
       2.8875        2.8875    0.0061835     0.017817

Current Coefficients:
         3.81544472859719
        0.145182166352519

Current Std Dev:
        0.095536985862006

Error = 8.13798271668736e-11

GradF1:
     -1.38276209327159e-06
     -6.38748018744195e-06
     -6.21835131005971e-06

Hessian:
 -36                  -116.4864              145.978272113953
-116.4864             -429.48715608          468.509148871197
 145.978272113953      468.509148871197      -674.494748297997
```

```
J =
0.095536985862006                          0      -0.364516089093255
                0          0.095536985862006      -0.013870266574236
                0                          0      -0.00912731566759714
V =
 0.450746091372439      -0.0653603524663376       0.0521534267916939
-0.0653603524663376      0.0190876683112434      -0.000887247950956868
 0.0521534267916938     -0.000887247950956856     0.0121536651934179

CovMatrx =
0.00209652138696284     -0.000573327948904799     -5.04169819527687e-06
-0.000573327948904799    0.000178908774495219      2.3123102359153e-06
-5.04169819527686e-06    2.31231023591529e-06      1.0124962187808e-06
```

LIST OF REFERENCES

De Valois, Russell L., De Valois, Karen K. (1988). Spatial Vision. Oxford Psychological Series No. 14. Oxford University Press, New York.

'Enhanced vision' goal: Fly anywhere, anytime. (1997, April 21). Aviation Week and Space Technology, 53.

Essock, E.A., McCarley, J.S., Sinai, M.J., and Krebs, W.K. (1996). Functional assessment of night-vision enhancement of real-world scenes. Investigative Ophthalmology and Visual Science, (SUPPL) 36, 2368, FT Lauderdale, FL.

Essock, E.A., Sinai, M.J., Srinivasan, N., DeFord, J.K., and Krebs, W.K. (1997). Texture-based segmentation in real world nighttime scenes. Investigative Ophthalmology and Visual Science, (SUPPL) 38, 2991, FT Lauderdale, FL.

Gescheider, G. A. (1985). Phychophysics: Method, Theory, and Application. Lawrence Erlbaum Associates, Inc..

Greene, William H. (1997). Econometric Analysis. Prentice Hall, Inc.

Hartline, P.H. and Newman, E.A. (1982). The Infrared "Vision" of Snakes. Scientific American, 246, 3.

Krebs, W.K., Elias, J.J., Parker, J.E., and Still, D.L. (1994). Using a fast Fourier transform (FFT) to identify differences between night vision goggle (NVG) images versus unaided images: How does this effect night attack? Proceedings of the Aerospace Medical Association 64th Annual Science Meeting Program, San Antonio, TX.

Krebs, W.K., Scribner, D.A., Schuler, J., Miller, G., and Lobik, D. (1996). Human factor test and evaluation of a low light sensor fusion device for automobile applications. Automotive Night Vision/Enhanced Driving Conference, Detroit, MI.

Macmillan, N.A., and Creelman, C. D. (1991). Detection Theory: A User's Guide. Cambridge University Press.

Nill, N.B., and Bouzas, B.H. (1991). Objective Image Quality Measure Derived from Digital Image Power Spectra. Optical Engineering, Vol. 31, No. 4, 1992, April.

Palmer, J., Ryan, D., Tinkler, R., Creswick, H. (1993). Assessment of image fusion in a night pilotage system. NATO AC/243 Panel 3/4 Symposium on Multisensors and Sensor Fusion, Brussels, Belgium.

Pelli, D. G., Rubin, G. S., and Legge, G. E. (1986). Predicting the Contrast Sensitivity of Low-Vision Observers. J. Opt. Soc. Amer. A, vol. 3.

Pelli, D. G., Rubin, G. S., and Legge, G. E. (1990). Clinical Measurement of Contrast Sensitivity. Investigative Ophalmology and Visual Science.

Pratt, W. K. (1991). Digital Image Processing. New York: John Wiley & Sons, Inc.

Russ, J. C. (1995). The Image Processing Handbook. Boca Raton: CRC Press

Sampson, M. T. (1996). An Assessment of the Impact of Fused Monochrome and Fused Color Night Vision Displays on Reaction Time and Accuracy in Target Detection. (M.S. in Operations Research) Monterey, CA: Naval Postgraduate School, September.

Sampson, M.T., Krebs, W.K., Scribner, D.A., and Essock, E.A. (1996). Visual search in natural (visible, infrared, and fused visible and infrared) stimuli. Investigative Ophthalmology and Visual Science, (SUPPL) 36, 1362, FT Lauderdale, FL.

Satyshur, M. P., Scribner, D. A., Kruer, M.R. (1997). Multispectral Imaging: Band Selection and Performance Predictions. 4 Feb, 1997.

Sekuler, R. and Blake, R. (1990). Perception. New York: McGraw-Hill.

Schiffman, H. R. (1990). Sensation and Perception: An Integrated Approach. New York: John Wiley & Sons.

Scribner, D., Satyshur, M. P., Kruer, M. R. (1993). Composite Infrared Color Images and Related Processing. Proceedings of the IRIS Specialty Group on Targets, Backgrounds, and Discrimination.

Scribner, D., Satyshur, M. P., Shuler, J. and Kruer, M. (1996). Infrared Color Vision. Proceedings of the IRIS IRIS Specialty Group on Targets, Backgrounds, and Discrimination.

Therrien, C.W., Scrofani, J., and Krebs, W.K. (1997). An adaptive technique for the enhanced fusion of low-light visible with uncooled thermal infrared imagery. IEEE: International Conference on Imaging Processing, October 1997.

Thomas, C. W., Gilmore, G. C., and Royer, F. L. (1992). Models of Contrast Sensitivity in Human Vision. IEEE: Transactions on Systems, Man, and Cybernetics, Vol. 23, No. 3, 1993, May/June.

Tolhurst, D. J., Tadmor, Y., and Chao, Tang (1991). Amplitude Spectra of Natural Images. Ophthal. Physiol. Opt., 1992., Vol. 12, April.

Waxman, A.M., Savoye, E.D., Fay, D.A., Aguilar, M., Gove, A.N., Carrick, J.E., Racamato, J.P. (1996). Electronic imaging aids for night driving: low-light CCD, uncooled thermal IR, and color fused visible/LWIR. Proceedings of the SPIE Conference on Transportation Sensors and Controls: Collision Avoidance, Traffic Management, and ITS, 2902, 62-73.

INITIAL DISTRIBUTION LIST

Number of Copies

1. Defense Technical Information Center 2
   8725 John J. Kingman Rd., Ste 0944
   Ft. Belvoir, Virginia 22060-6218

2. Dudley Knox Library 2
   Naval Postgraduate School
   411 Dyer Rd.
   Monterey, CA 93943-5101

3. William Krebs, Code OR/Kw 3
   Department of Operations Research
   Naval Postgraduate School
   Monterey, CA 93943-5101

4. James Eagle, Code OR/Er 1
   Department of Operations Research
   Naval Postgraduate School
   Monterey, CA 93943-5101

5. Dean Scribner 1
   Naval Research Laboratory
   Optical Sciences Division/Code 5630
   4555 Overlook Avenue
   Washington D. C., 20375-5000

6. Robert McDaniel 1
   Lockheed Martin Electronics and Missiles
   5600 Sand Lake Road, MP 126
   Orlando, Florida 32819-8907

7. Joel Davis 1
   Office of Naval Research
   Physiology/Neurobiology Division
   800 North Quincy Street
   Arlington, Virginia 22217-5660